PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-299641
(43)Date of publication of application : 24.10.2000

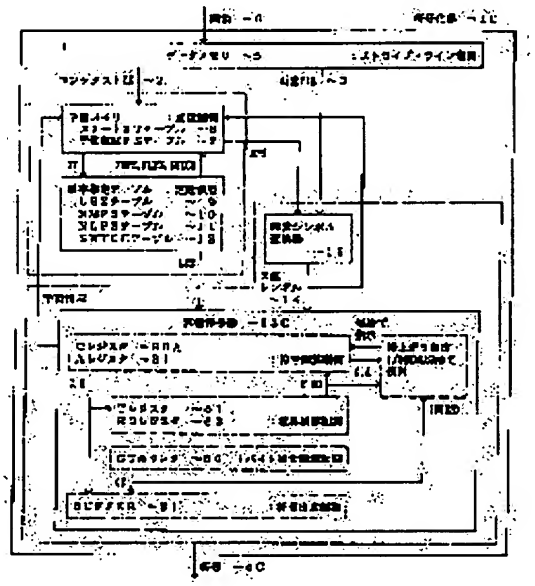
(51)Int.Cl. H03M 7/40
G06F 5/00
H04N 1/413
H04N 7/24

(21)Application number : 11-104162
(22)Date of filing : 12.04.1999

(71)Applicant : MITSUBISHI ELECTRIC CORP
(72)Inventor : KIMURA TOMOHIRO
YOSHIDA MASAYUKI
ONO FUMITAKA

(54) ENCODER, DECODER, DEVICE FOR ENCODING AND DECODING, ENCODING METHOD AND DECODING METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To reduce a total code length and a code output time by allowing an encoder to ignore one of partial areas obtained by equally dividing an effective area indicated by a carry detector to a high order and a low order so as to update the effective area.
SOLUTION: An arithmetic encoder 13C executes encoding arithmetic by means of a C register 30A and an A register 31 from the inputs of an LSZ value 9 and a binary symbol 14, measures the code outputting timing of a byte unit by a CT counter 50 and judges the presence/absence of a carry boundary in an area from a T register 61 and an R0 register 63 synchronizing to the time of outputting a code. Then, when the carry boundary exists, the encoder 13C forcedly defines the presence/absence of carry by correcting the C register 30A and the A register 31 by applying a 1/2 area ignoring system, executes carry propagation until a maximum BUFFER 51 to execute defined encoding output, and outputs a code 4C. Thus, one of partial areas obtained by equally dividing an effective area indicated by a carry detector to a high order and a low order like this is ignored to update the effective area.



LEGAL STATUS

[Date of request for examination]	17.06.2003
[Date of sending the examiner's decision of rejection]	
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]	
[Date of final disposal for application]	
[Patent number]	3745160
[Date of registration]	02.12.2005
[Number of appeal against examiner's decision of rejection]	
[Date of requesting appeal against examiner's decision of rejection]	
[Date of extinction of right]	

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2000－299641
（Ｐ２０００－２９９６４１Ａ）
(43)公開日　平成12年10月24日(2000.10.24)

(51)Int.Cl.[7]	識別記号	Ｆ Ｉ		テーマコード（参考）
H03M 7/40		H03M 7/40		5C059
G06F 5/00		G06F 5/00	H	5C078
H04N 1/413		H04N 1/413	Z	5J064
7/24		7/13	Z	9A001

審査請求　未請求　請求項の数29　ＯＬ　（全36頁）

(21)出願番号　特願平11－104162

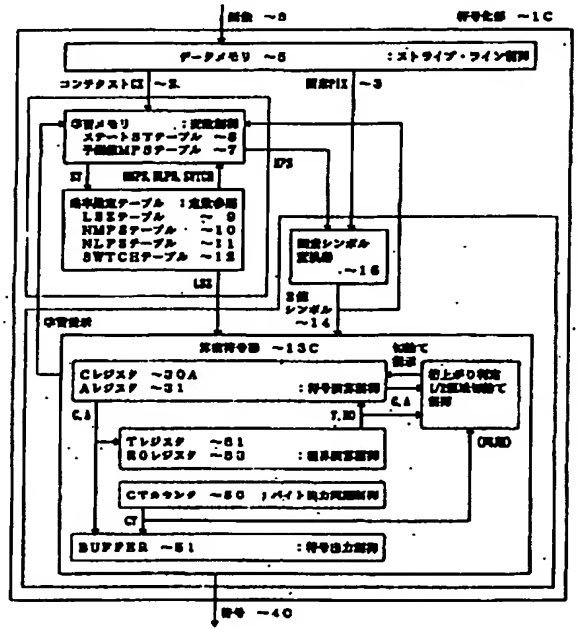
(22)出願日　平成11年４月12日(1999.4.12)

(71)出願人　000006013
三菱電機株式会社
東京都千代田区丸の内二丁目２番３号
(72)発明者　木村　智広
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
(72)発明者　吉田　雅之
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内
(74)代理人　100057874
弁理士　曾我　道照　（外６名）

最終頁に続く

(54)【発明の名称】符号化装置、復号化装置、符号化復号化装置、符号化方法並びに復号化方法

(57)【要約】
【課題】　桁上がりを制御により符号を確定させ所定のパターンへの置き換えにより総符号長と符号の掃出し時間を短縮する。
【課題手段】　情報源データを蓄積するデータメモリ、補助パラメータを基に指定される符号化対象データに関する学習データを蓄積する学習メモリ、学習データを基に指定される符号化パラメータを出力する確率推定テーブル、符号化対象データと符号化パラメータを基に算術符号化を行い符号を出力する符号器を備える符号化装置において、情報源データの入力処理毎または符号の出力毎に所定の間隔を計測して通知する同期検出器、所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器を備え、符号器は、桁上がり検出器により指示された有効領域を上位と下位に等分した部分領域の一方を切り捨てて有効領域を更新する。



1

【特許請求の範囲】

【請求項１】　情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータメモリと、上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される符号化パラメータを出力する確率推定テーブルと、上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号器とを備える符号化装置において、

所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出器と、

上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器と

を備え、上記符号器は、上記桁上がり検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とする符号化装置。

【請求項２】　上記符号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記桁上がり境界検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とする請求項１に記載の符号化装置。

【請求項３】　上記桁上がり境界検出器は、検出された桁上がり境界を含む部分領域を切り捨てるように指示することを特徴とする請求項１または２に記載の符号化装置。

【請求項４】　上記所定の間隔は、所定単位の符号化対象データを符号化したときであることを特徴とする請求項１または２に記載の符号化装置。

【請求項５】　上記所定の間隔は、符号値の確定まで待機させる符号長の最大値を基準に設定されることを特徴とする請求項４に記載の符号化装置。

【請求項６】　上記符号器は、上記所定の間隔を基準に符号値の確定まで待機させる符号長の最大値が設定されることを特徴とする請求項４に記載の符号化装置。

【請求項７】　上記所定の間隔は、所定単位の符号出力が発生したときであることを特徴とする請求項１または２に記載の符号化装置。

【請求項８】　上記所定の間隔は、所定単位の符号出力が発生し、かつ符号出力値が所定値となるときであることを特徴とする請求項１または２に記載の符号化装置。

【請求項９】　上記桁上がり境界検出器は、上記有効領域内に桁上がり境界が検出されなくても上記有効領域を上位と下位に等分した部分領域のどちらか一方の部分領域を指示することを特徴とする請求項１に記載の符号化装置。

【請求項１０】　上記符号器は、確定した符号に同じ値が連続するときにその値と長さを特定できるランレングスマーカに置き換えるランレングスマーカ変換器を有することを特徴とする請求項１または２に記載の符号化装置。

2

【請求項１１】　上記ランレングスマーカ変換器は、置き換えずにそのまま出力した方が変換される上記ランレングスマーカより符号出力が短いときには変換を行わないことを特徴とする請求項１０に記載の符号化装置。

【請求項１２】　上記ランレングスマーカから抽出される連続長の最大値は、復号の際のランレングスマーカ逆変換の結果に基づいて上記符号器が上記ランレングスマーカ変換器に設定することを特徴とする請求項１０に記載の符号化装置。

【請求項１３】　復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータメモリと、上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される復号パラメータを出力する確率推定テーブルと、上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号器とを備える復号化装置において、

所定の単位の情報源データの出力処理または符号の入力を所定の間隔として計測して通知する同期検出器と、

上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、上記復号器は、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とする復号化装置。

【請求項１４】　上記復号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とする請求項１３に記載の復号化装置。

【請求項１５】　上記桁上がり境界検出器は、検出された桁上がり境界を含む部分領域を切り捨てるように指示することを特徴とする請求項１３または１４に記載の復号化装置。

【請求項１６】　上記所定の間隔は、所定単位の復号対象データを復号したときであることを特徴とする請求項１３または１４に記載される復号化装置。

【請求項１７】　上記所定の間隔は、所定単位の符号入力が発生したときであることを特徴とする請求項１３または１４に記載される復号化装置。

【請求項１８】　上記所定の間隔は、所定単位の符号入力が発生し、かつ復号と同時に再現される符号出力値が所定値となるときであることを特徴とする請求項１３に記載の復号化装置。

【請求項１９】　上記復号器は、ランレングスマーカを検出すると、置き換えられた長さの元の符号値に逆変換

するランレングスマーカ逆変換器を有することを特徴とする請求項１３または１４に記載される復号化装置。

【請求項２０】　符号化装置として、情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータメモリと、上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される符号化パラメータを出力する確率推定テーブルと、上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号器と、所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、

上記符号器は、上記桁上がり検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新すると共に、

復号化装置として、復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータメモリと、上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される復号パラメータを出力する確率推定テーブルと、上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号器と、所定の単位の情報源データの出力処理または符号の入力を所定の間隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、

上記復号器は、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とする符号化復号化装置。

【請求項２１】　上記符号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記桁上がり境界検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新すると共に、上記復号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とする請求項２０に記載の符号化復号化装置。

【請求項２２】（ａ）情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータ蓄積ステップと、（ｂ）上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習ステップと、（ｃ）上記学習データを基に指定される符号化パラメータを出力する確率推定ステップと、（ｄ）上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号化ステップと、（ｅ）所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出ステップと、（ｆ）上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出ステップと、（ｇ）上記桁上がり境界検出ステップにより指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新する符号化修正ステップとを有する符号化方法。

【請求項２３】　上記符号化修正ステップは、上記桁上がり境界検出ステップで桁上がり境界が検出されたとき、上記桁上がり境界検出ステップで指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とする請求項２２に記載の符号化方法。

【請求項２４】　上記符号化ステップおよび上記符号化修正ステップは、確定した符号に同じ値が連続するときにその値と長さを特定できるランレングスマーカに置き換える変換ステップを有することを特徴とする請求項２２または２３に記載の符号化方法。

【請求項２５】　上記変換ステップは、置き換えずにそのまま出力した方が変換される上記ランレングスマーカより符号出力が短いときには変換を行わないことを特徴とする請求項２４に記載の符号化方法。

【請求項２６】　上記ランレングスマーカから抽出される連続長の最大値は、復号の際、ランレングスマーカ逆変換の結果に基づいて上記符号化ステップが上記変換ステップに設定することを特徴とする請求項２４に記載の符号化方法。

【請求項２７】（ａ）復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータ蓄積ステップと、（ｂ）上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習ステップと、（ｃ）上記学習データを基に指定される復号パラメータを出力する確率推定ステップと、（ｄ）上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号化ステップと、（ｅ）所定の単位の情報源データの出力処理または符号の入力を所定の間隔として計測して通知する同期検出ステップと、（ｆ）上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出ステップと、（ｇ）上記桁上がり境界検出ステップにより指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新する復号化修正ステップとを有する復号化方法。

【請求項２８】　上記復号化修正ステップは、上記桁上がり境界検出ステップで桁上がり境界が検出されたとき、上記桁上がり境界検出ステップで指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とする請求項２７に記載の復号化方法。

【請求項２９】　上記復号化ステップは、ランレングスマーカを検出すると、置き換えられた長さの元の符号値に逆変換するランレングスマーカ逆変換ステップを有することを特徴とする請求項２７または２８に記載の復号化方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は、画像あるいはデータの算術符号化および算術復号化に係るもので、符号化装置、復号化装置、符号化復号化装置、符号化方法並びに復号化方法に関するものである。

【０００２】

【従来の技術】図３９は算術符号化の概念を示す説明図である。図３９において、数直線上の全体の有効領域を優勢シンボルＭＰＳと劣勢シンボルＬＰＳに対する二つの部分領域に分割し、出現したシンボルに対応する部分領域を新規有効領域に更新するという手順で符号化が行われる。しかし、領域幅が小さくなるにつれて２進小数（固定小数点）で示したように下界値、上界値は有効桁数が増加し、演算のためのレジスタ精度も必要となってくる。

【０００３】ここで、有効桁数が増加しても上位桁はほとんど変化がないことに注目し、図４０に示すように、小数部を常に領域分割の精度（図では３桁）に保ち、上位桁を整数部に掃き出すために正規化処理を行う。図４０では、正規化処理で２のべき乗倍に拡大することによって、有効領域幅Ａを１／２に当たる４以上、１に当たる８以下（８は初期値のみ）にとし、下界値（符号）Ｃとすれば有効領域はＣ以上Ｃ＋Ａ未満で表すことができる。

【０００４】第１の従来例について

第１の従来例の算術符号化の符号器（Encoder）・復号器（Decoder）は、ＩＴＵ－Ｔの国際標準勧告Ｔ．８２に記載されるテーブルおよび処理を説明するフローチャートによって実現できる。以下、この算術符号化をＱＭ－Ｃｏｄｅｒ、その符号器を符号部１Ａ、その復号器を復号部１６Ａと呼ぶものとし、その符号器と復号器の概略構成を図４１、図４２に示す。

【０００５】図４１において、符号部１Ａの入力は２つあり、第１の入力はコンテクストＣＸ（Context）２であり、第２の入力は符号化する画素ＰＩＸ（Pixel）３である。出力は符号（Code）４Ａである。また、図４２において、復号部１６Ａの入力は２つあり、第１の入力はコンテクストＣＸ２であり、第２の入力は符号４Ａである。出力は、復号された画素ＰＩＸ３である。

【０００６】データメモリ５は、画像（Image）６を蓄積し、符号化または復号化の対象となる画素に対して共通に参照できる近隣のすでに符号化／復号化済みの画素の中で、参照位置をモデルテンプレート（Model Template）で指定される１０画素の参照パターンであるコンテクスト（１０ビット。総数１０２４）ＣＸ２を生成する。また、符号化のときは同時に符号化対象画素を出力し、復号化のときは復号された復号対象画素を蓄積する。

【０００７】ＱＭ－Ｃｏｄｅｒには、コンテクストＣＸ２を生成するための図４３に示される２ラインおよび３ラインの標準モデルテンプレートのいずれかを選択し、規定のヘッダ情報によって符号器から復号器に通知する。

【０００８】ＱＭ－Ｃｏｄｅｒでは、符号化／復号化対象画素に対する各コンテクストごとに画素値の予測一致確率を推定し、その変動に伴って学習しながら符号化／復号化を進める。学習は、コンテクストをインデクスとする２つの変数テーブルで構成される学習メモリを書き換えることによって行われる。１つは、出現する確率の高い画素値ＭＰＳ（More Probable Symbol：優勢シンボル）を予測値として記憶する各１ビットのＭＰＳテーブル７である（予測一致確率の低い画素値はＬＰＳ［Less Probable Symbol：劣勢シンボル］という）。もう１つは、予測値の予測一致確率の程度を総計１１３の状態（State）に分類した状態番号（０～１１２）を記憶する各７ビットのＳＴテーブル８である。初期値はＭＰＳテーブル７、ＳＴテーブル８ともに全て０である。

【０００９】変数テーブルの他に、符号化／復号の際に状態番号（ステート）をインデクスとして参照する４つの定数テーブルで構成される確率推定テーブル（Probability Estimation Table）が存在する。ＬＰＳ領域幅を１６ビットで表すＬＳＺテーブル９、ＭＰＳ遷移先を７ビットで表すＮＭＰＳテーブル１０、ＬＰＳ遷移先を７ビットで表すＮＬＰＳテーブル１１、予測値反転判定を１ビットで表すＳＷＴＣＨテーブル１２である。その設定値を図４４に示す（ここで示した英字の変数・定数テーブル名は、後述の処理を説明するフローチャートで用いる配列名とする）。

【００１０】ＬＳＺテーブル９は、算術符号器１３Ａ／算術復号器１７Ａ内の演算部で参照され、適応予測の学習には直接的には関係しない。算術符号器１３Ａ／算術復号器１７Ａの内部ではＬＳＺ値を用いて演算が行われ、演算精度が低下した際には正規化処理（Renormalization）を行う。この正規化処理が行われるとき、同時に学習が指示される。

【００１１】符号化のとき、画素シンボル変換器１５は、コンテクストＣＸ２をインデクスとしてＭＰＳテーブル７から出力されたＭＰＳ値と画素ＰＩＸ３の一致／

不一致を判定して、２値シンボル１４を算術符号器１３Ａ、学習メモリ（ＭＰＳテーブル７、ＳＴテーブル８）へ出力し、学習指示により学習を行う。

【００１２】また、復号のとき、シンボル画素変換器１８は、算術復号器１７Ａで復号された２値シンボル１４から、コンテクストＣＸ２をインデクスとしてＭＰＳテーブル７から出力されたＭＰＳ値と復号すべき画素ＰＩＸ３の一致／不一致を判定し、一致ならばＭＰＳ値、不一致ならばＬＰＳ値（＝１－ＭＰＳ）を画素ＰＩＸ３値ととして復号する。復号された２値シンボル１４は、学習メモリ（ＭＰＳテーブル７、ＳＴテーブル８）へ出力し、学習指示により学習を行う。

【００１３】学習が指示されたとき、符号化／復号対象の２値シンボル１４が優勢シンボルであればＮＭＰＳテーブル値１０を、劣勢シンボルであればＮＬＰＳテーブル値１１をＳＴテーブル８に書き込み、状態遷移が実現される。また、劣勢シンボルによる学習では、その予測一致確率が１／２であればＭＰＳテーブル値を反転（演算「１－ＭＰＳ」）させてＭＰＳテーブルに書き込む。その一致確率が１／２であるかどうかは、ＳＷＴＣＨ値をフラグとして判定される。

【００１４】算術符号器１３Ａは、ＬＳＺ値９と２値シンボル１４の入力から、符号（領域下界値）を示すＣレジスタ３０Ａと領域幅を示すＡレジスタ３１で符号化演算が進められ、ＣＴカウンタ５０でバイト単位の符号出力タイミングを計り、ＢＵＦＦＥＲ５１とＳＣカウンタで桁上がりの伝搬可能性のある未確定の符号出力待機と確定した符号出力を行い、符号４Ａを出力する。

【００１５】また、算術復号器１７Ａは、ＬＳＺ値9と符号４Ａの入力から、ＣＴカウンタ５０でバイト単位の符号入力タイミングを計り、ＢＵＦＦＥＲ５１を介して、領域下界値から符号４Ａまでの変位を示すＣレジスタ３０Ｂと、領域幅を示すＡレジスタ３１で復号演算が進められ、２値シンボル１４を出力する。

【００１６】図４１と図４２に示される構成ブロック内およびブロック間の詳細な処理動作は、後述の処理を説明するフローを示す図４６ないし図６２で説明する。

【００１７】符号化処理を説明するフローおよび復号処理を説明するフローを説明する前に、符号化レジスタ（Ｃレジスタ）３０Ａ、復号レジスタ（Ｃレジスタ）３０Ｂおよび領域幅レジスタ（Ａレジスタ）３１のビット配置を図４５に示す。

【００１８】符号化レジスタ（Ｃ）３０Ａにおいて、ｂｉｔ１５とｂｉｔ１６の間に小数点を設定し、"ｘ"（１６ビット）はＬＳＺ９に対する演算部Ｃｘ３２であり、キャリがある場合はより上位へ伝搬する。"ｓ"（３ビット）はスペーサビット部Ｃｓ３３、"ｂ"（８ビット）はバイト出力部Ｃｂ３４、"ｃ"（１ビット）は桁上がり判定部Ｃｃ３５である。符号化の過程において、Ｃレジスタ値は符号４として符号化したシンボルに対応させた領域の下界値となるように更新される。

【００１９】復号レジスタ（Ｃ）３０Ｂにおいて、下位ワードＣＬＯＷ３６と上位ワードＣＨＩＧＨ３８は３２ビットのレジスタとして実現でき、ＭＳＢであるｂｉｔ３１の上位に小数点を設定し、"ｂ"（８ビット）はバイト入力部（ＣＬＯＷレジスタ３６）の上位バイトＣｂ３７、"ｘ"（１６ビット）はＬＳＺ９に対する演算部Ｃｘ（ＣＨＩＧＨレジスタ３８）３９である。復号の過程において、Ｃレジスタ値は復号したシンボルに対応させた領域の下界値からその領域内の座標である符号４へのオフセット値となるように更新される。

【００２０】領域幅レジスタ（Ａ）３１は符号化／復号で共通とし、符号化／復号レジスタ３０Ａ、３０Ｂの小数点に対応して、"ｘ"レジスタ部に合わせて"ａ"（１６ビット）が小数部として配置され、初期状態の値でのみ整数部（ｂｉｔ１６）が"１"なる。領域幅（領域サイズともいう）は、Ａ－ＬＳＺ（下方領域幅）またはＬＳＺ（上方領域幅）に更新され、初期値（整数部＝"１"）を除いて１／２の重みを示すｂｉｔ１５が"１"となるように正規化され、１／２以上に保つことによって、上方領域幅としていかなるＬＳＺ９を選択しても下方領域の確保を保証する。正規化処理では、Ａレジスタ３１、Ｃレジスタ３０Ａまたは３０Ｂを同時にシフト処理によって拡大する。

【００２１】ＱＭ－Ｃｏｄｅｒでは、状態に対して固定サイズとなる上方領域ＬＳＺ９は通常劣勢シンボルに割り当てられるが、下方領域が上方領域より小さくなるときには優勢シンボルに割り当てる「条件付きＭＰＳ／ＬＰＳ交換」を行っている。劣勢シンボルを符号化／復号したとき、および条件付きＭＰＳ／ＬＰＳ交換を適用して優勢シンボルを符号化／復号したときには正規化処理が必ず行われる。

【００２２】このレジスタのビット配置を基に、符号化の処理を説明するフロー、復号の処理を説明するフローを説明する。処理を説明するフロー中の用語「レイヤ」は階層的符号化の場合の「（解像度）層」、「ストライプ」は画像をＮライン単位（最終ストライプのみＮライン以下、また、Ｎ＝１でも構わない）で区切った「短冊」を意味する。ここでは、レイヤ数＝１、ストライプ＝ライン（Ｎ＝１）として説明するが、複数レイヤの符号化／復号への拡張を妨げるものではない。

【００２３】符号化／復号処理を説明するフローを説明するために、図４１、図４２、図４５で説明した変数、テーブル、レジスタの他に、次の補助変数ＣＴ５０、ＢＵＦＦＥＲ５１、ＳＣ５２、ＴＥＭＰ５３を使用する。ＣＴ５０は、正規化処理によるＣレジスタ３０Ａ、３０ＢおよびＡレジスタ３１のシフト数を計数し、値が０となったとき次の符号バイト入出力を行うための補助変数である。ＢＵＦＦＥＲ５１は、符号化のときＣレジスタ３０Ａから出力された符号バイト値、復号のときＣレジ

スタ３０Ｂへ入力される符号バイト値を格納する補助変数である。ＳＣ５２は、符号化のみで使用され、Ｃレジスタ３０Ａから出力された符号バイト値が０ｘＦＦであるときその連続数を計数する補助変数である。

【００２４】なお、本明細書において"０ｘ"で始まる数値は１６進数を表すものとする。ＴＥＭＰ５３は、符号化のみで使用され、ＢＵＦＦＥＲ５１への桁上がりを検出し、桁上がり処理後はその下位８ビットを新たなＢＵＦＦＥＲ値５１とするための補助変数である。Ｃレジスタ３０ＡからＴＥＭＰ５３を介して設定されるＢＵＦＦＥＲ５１は桁上がり処理を行わずに０ｘＦＦとなることはなく、その時点でＢＵＦＦＥＲ５１から下位、つまりＢＵＦＦＥＲ５１、ＳＣ５２個の０ｘＦＦはＣレジスタ３０Ａより上位への桁上がりあれば変更される可能性があるためＣレジスタ３０Ａから出力されていても符号４として確定できていない。

【００２５】図４６は、符号化処理の全体の流れを示すＥＮＣＯＤＥＲ処理を説明するフローチャートである。国際標準勧告Ｔ．８２の本処理を説明するフローチャートにあるＴＰ（Typical Prediction）およびＤＰ（Deterministic Prediction）に関する処理は、直接的な関係がないため省略する。まず、Ｓ１０１ではＩＮＩＴＥＮＣ処理を呼び出し、符号化処理の初期化を行う。Ｓ１０２では画素ＰＩＸ３とコンテクストＣＸ２の組を１つずつ読み出し、Ｓ１０３ではＥＮＣＯＤＥ処理によって符号化を行う。Ｓ１０４ではストライプ（ここではライン）が終了するまでＳ１０２およびＳ１０３を繰り返し、さらにＳ１０５では画像が終了するまでＳ１０２からＳ１０４までストライプの符号化を繰り返す。Ｓ１０６でＦＬＵＳＨ処理を呼び出して符号化処理の後処理を行う。

【００２６】図４７は、符号化対象画素値３と予測値７の一致・不一致から呼び出す処理を切り替えるＥＮＣＯＤＥ処理を説明するフローチャートである。Ｓ１１１では画素値３と予測値７の一致・不一致を判定し、一致ならば優勢シンボル、不一致ならば劣勢シンボルを符号化する。Ｓ１１３ではＣＯＤＥＭＰＳ処理を呼び出して優勢シンボルを、Ｓ１１２ではＣＯＤＥＬＰＳ処理を呼び出して劣勢シンボルを符号化する。

【００２７】図４８は、符号化対象画素値３と予測値７が不一致のとき、すなわち劣勢シンボルを符号化する際に呼び出されるＣＯＤＥＬＰＳ処理を説明するフローチャートである。まず、Ｓ１２１では一時的にＡレジスタ３１値は、下方領域幅に更新される。Ｓ１２２の判定がＹｅｓならば、条件付きＭＰＳ／ＬＰＳ交換が適用され、下方領域を符号化するためＡレジスタ３１はそのままとし、Ｃレジスタ３０Ａは更新しない。Ｓ１２２の判定がＮｏならば、上方領域を符号化し、Ｓ１２３で領域下界値であるＣレジスタ３０Ａ、Ｓ１２４で領域幅であるＡレジスタ３１を更新する。Ｓ１２５の判定で定数ＳＷＴＣＨ値１２が１ならば、Ｓ１２６で予測値（ＭＰＳテーブル）の反転・更新を行う。ＬＰＳ符号化では、Ｓ１２７でＮＬＰＳテーブル１１参照による状態遷移を行い、Ｓ１２８でＲＥＮＯＲＭＥ処理を呼び出して正規化処理を行う。

【００２８】図４９は、符号化対象画素値３と予測値７が一致のとき、すなわち優勢シンボルを符号化する際に呼び出されるＣＯＤＥＭＰＳ処理を説明するフローチャートである。まず、Ｓ１３１では一時的にＡレジスタ３１値は、下方領域幅に更新される。Ｓ１３２の判定がＮｏならばそのままＣＯＤＥＭＰＳ処理を終了する。Ｓ１３２の判定がＹｅｓならば、必ずＳ１３６でＮＭＰＳテーブル１０参照による状態遷移を行い、Ｓ１３７でＲＥＮＯＲＭＥ処理を呼び出して正規化処理を行う。Ｓ１３６およびＳ１３７の前に、Ｓ１３３の判定がＹｅｓならば、下方領域を符号化するためＡレジスタ３１はそのままとし、Ｃレジスタ３０Ａは更新しない。Ｓ１３３の判定がＮｏならば、条件付きＭＰＳ／ＬＰＳ交換が適用され、上方領域を符号化し、Ｓ１３４でＣレジスタ３０Ａ、Ｓ１３５でＡレジスタ３１を更新する。

【００２９】図５０は、正規化処理を行うＲＥＮＯＲＭＥ処理を説明するフローチャートである。Ｓ１４１ではＡレジスタ３１、Ｓ１４２ではＣレジスタ３０Ａを１ビット上位へシフトすることによって２倍の乗算と等価な演算を行う。Ｓ１４３で変数ＣＴ５０から１を減じ、Ｓ１４４で変数ＣＴ５０値が０か否かを判定し、判定がＹｅｓならばＳ１４５でＢＹＴＥＯＵＴ処理を呼び出してＣレジスタ３０Ａから１バイト符号４を出力する。Ｓ１４６の判定は、正規化処理の終了判定を行っており、Ａレジスタ３１が０ｘ８０００未満ならばＳ１４１からＳ１４５を繰り返し、０ｘ８０００以上となれば終了する。

【００３０】図５１は、Ｃレジスタ３０Ａから１バイトずつ符号４を出力させるＢＹＴＥＯＵＴ処理を説明するフローチャートである。出力するバイトは符号化レジスタＣ３０Ａのバイト出力部Ｃｂ３４である。桁上がりを判定するための桁上がり判定部Ｃｃ３５も同時に処理する。Ｓ１５１でＣｂレジスタ３４およびＣｃレジスタ３５の計９ビットを変数ＴＥＭＰ５３に設定する。Ｓ１５２で桁上がりがある場合（ＴＥＭＰ≧０ｘ１００；Ｃｃ＝１）、桁上がりがない場合にはＳ１５９で０ｘＦＦである場合、０ｘＦＦ未満である場合に分けてバイト出力を処理する。Ｓ１５２の判定がＹｅｓの場合、すでにＣレジスタ３０Ａから出力済みの変数ＢＵＦＦＥＲ５１値にＳ１５３で桁上がりの１を加えた値、Ｓ１５４で変数値ＳＣ５２個のバイト値０（蓄積していた０ｘＦＦが桁上がりの伝搬により０ｘ００となった）の合計ＳＣ＋１バイトが桁上がりを伝搬された符号値として確定する。

【００３１】Ｓ１５５で変数ＳＣ５２を０とし、Ｓ１５６で変数ＢＵＦＦＥＲ５１に変数ＴＥＭＰの下位８ビッ

トを設定する。Ｓ１５７で変数ＴＥＭＰ５３として処理したＣｃレジスタ３５、Ｃｂレジスタ３４をクリアし、Ｓ１５８で次の出力まで８ビットを処理するため変数ＣＴ５０に８を設定する。Ｓ１５９の判定がＹｅｓの場合は、符号４は確定できず、変数値ＳＣ５２に１を加えて０ｘＦＦを蓄積する。Ｓ１５９の判定がＮｏの場合は、すでにＣレジスタ３０Ａから出力済みの符号４をＳ１５３でＢＵＦＦＥＲ５１値、Ｓ１５４で変数値ＳＣ５２個のバイト値０ｘＦＦの合計ＳＣ＋１バイトが符号値として確定する。Ｓ１６３で変数ＳＣ５２を０とし、Ｓ１６４で変数ＢＵＦＦＥＲ５１に変数ＴＥＭＰ５３（桁上がりがないためそのまま８ビット）を設定する。

【００３２】図５２は、符号化開始時のＳＴテーブル８、ＭＰＳテーブル７および各変数の初期値を設定するＩＮＩＴＥＮＣ処理を説明するフローチャートである。Ｓ１７１の「このレイヤの第１ストライプ」はレイヤおよびストライプの概念を持ち込まない場合「画像の符号化開始の時点」を意味し、複数のストライプから構成される画像では変数テーブルをストライプごとに初期化せずに処理を続けることもできる。Ｓ１７１では、このレイヤの画素の第１ストライプか、またはテーブルを強制リセットするのかを判定する。

【００３３】Ｓ１７１の判定がＹｅｓならば、Ｓ１７２ですべてのコンテクストＣＸ２に対して変数テーブルＳＴ８、ＭＰＳ７を初期化する。Ｓ１７３はＳＣ５２を、Ｓ１７４はＡレジスタ３１値を、Ｓ１７５はＣレジスタ３０Ａを、Ｓ１７６は変数ＣＴ５０を初期化する。変数ＣＴ５０の初期値１１はＣｂレジスタ３４のビット数とＣｓレジスタ３３のビット数の和であり、１１ビット処理したとき最初の符号出力を行うことになる。Ｓ１７１の判定がＮｏの場合は、Ｓ１７７では変数テーブルの初期化を行わず、同じレイヤの直前のストライプ終端のテーブル値を再設定する。

【００３４】図５３は、符号化終了時に符号化レジスタ３０Ａに残った値を掃き出す処理を含めた後処理を行うＦＬＵＳＨ処理を説明するフローチャートである。Ｓ１８１でＣＬＥＡＲＢＩＴＳ処理を呼び出してＣレジスタ３０Ａに残った符号の有効桁数を最小にする。Ｓ１８２はＦＩＮＡＬＷＲＩＴＥＳ処理を呼び出して変数ＢＵＦＦＥＲ５１、ＳＣ５２およびＣレジスタ３０Ａの未確定の符号４を確定して最終的に出力する。Ｓ１８３で符号４の第１バイトは変数ＢＵＦＦＥＲ５１がＣレジスタ３０Ａ値の出力に先だって（整数部として）出力されるため除去する。Ｓ１８４で符号４は最終有効領域内の小数座標であるから、必要であれば終端に連続するバイト０ｘ００は除去する。

【００３５】図５４は、符号化終了時の符号４の有効桁数を最小とするための処理を行うＣＬＥＡＲＢＩＴＳ処理を説明するフローチャートである。これによって、符号４は終端に可能な限り０ｘ００が連続する値となる。Ｓ１９１は変数ＴＥＭＰ５３として最終有効領域の上界値の下位２バイト（Ｃｘレジスタ３２）をクリアした値を設定している。Ｓ１９２では上界値の下位２バイトのクリアした値とＣレジスタ３０Ａ値との大小を比較する。Ｓ１９２の判定がＹｅｓならば、Ｓ１９３で変数値ＴＥＭＰ５３にクリアし過ぎた分を戻してＣレジスタ３０Ａ値とする。Ｓ１９２の判定がＮｏならば、変数値ＴＥＭＰ５３をＣレジスタ３０Ａ値とする。

【００３６】図５５は、符号化終了時点で確定した符号をＣレジスタ３０Ａに残った値まで書き出すＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフローチャートである。Ｓ２０１でＣレジスタを変数値ＣＴ５０で示されるビット数だけシフトして符号出力および桁上がり判定を可能とする。Ｓ２０２で桁上がりの有無を判定する。Ｓ２０２の判定がＹｅｓならば桁上がりがあり、Ｎｏならば桁上がりはない。ＢＹＴＥＯＵＴ処理を説明するフローと同様にＳ２０３、Ｓ２０４で桁上がりのある符号値、Ｓ２０７、Ｓ２０８で桁上がりのない符号値とするＣレジスタから出力済みのＳＣ＋１バイトの符号４を確定する。Ｓ２０５でＣｂレジスタ値（１バイト）、Ｓ２０６でその下位１バイトを出力して符号出力が完了する。

【００３７】図５６は、復号処理の全体の流れを示すＤＥＣＯＤＥＲ処理を説明するフローチャートである。国際標準勧告Ｔ．８２の本処理を説明するフローチャートにあるＴＰおよびＤＰに関する処理は本発明および従来例（第１の従来例および第２の従来例）とは関係がないため省略する。まず、Ｓ２１１ではＩＮＩＴＤＥＣ処理を呼び出し、復号処理の初期化を行う。Ｓ２１２ではコンテクストＣＸ２を１つずつ読み出し、Ｓ２１３ではＤＥＣＯＤＥ処理によって画素ＰＩＸ３の復号を行う。Ｓ２１４ではストライプ（ここではライン）が終了するまでＳ１１２およびＳ１１３を繰り返し、さらにＳ２１５では画像が終了するまでＳ２１２からＳ２１４までストライプの復号を繰り返す。

【００３８】図５７は、復号対象画素を復号するＤＥＣＯＤＥ処理を説明するフローチャートである。まず、Ｓ２２１で一時的にＡレジスタ３１値は、下方領域幅に更新される。Ｓ２２２の判定がＹｅｓならば、下方領域を復号する。Ｓ２２３の判定がＹｅｓならば、Ｓ２２４でＭＰＳ＿ＥＸＣＨＡＮＧＥ処理を呼び出して、Ｓ２２５でＲＥＮＯＲＭＤ処理を呼び出して正規化処理を行う。Ｓ２２３の判定がＮｏならば、正規化処理することなく優勢シンボルを復号し、予測値７を画素値３とする。また、Ｓ２２２の判定がＮｏならば、上方領域を復号する。Ｓ２２７でＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を呼び出して、Ｓ２２８でＲＥＮＯＲＭＤ処理を呼び出して正規化処理を行う。ＭＰＳ＿ＥＸＣＨＡＮＧＥ処理、ＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を呼び出すパスでは、それぞれ復号すべき領域は決まっていても領域の大小を比較しないと復号対象シンボルが優勢シンボルか劣勢シンボルか

判断できない。呼び出した処理を説明するフローチャートでそれぞれ復号される画素値３を決定する。

【００３９】図５８は、上方領域を復号するＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。

Ｓ２３１の判定がＹｅｓならば優勢シンボルを復号する。Ｓ２３２でＣレジスタ３０Ｂを、Ｓ２３３でＡレジスタ３１を更新する。Ｓ２３４では予測値７をそのまま画素値３とする。Ｓ２３５でＮＭＰＳテーブル１０参照による状態遷移を行う。また、Ｓ２３１の判定がＮｏならば劣勢シンボルを復号する。Ｓ２３６でＣレジスタ３０Ｂを、Ｓ２３７でＡレジスタ３１を更新する。Ｓ２３８では非予測値「１－予測値」を画素値３とする。Ｓ２３９の判定がＹｅｓならば、Ｓ２４０で予測値（ＭＰＳテーブル）７の反転・更新を行う。Ｓ２４１でＮＬＰＳテーブル１１参照による状態遷移を行う。

【００４０】図５９は、下方領域を復号するＭＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。

Ｓ２５１の判定がＹｅｓならば劣勢シンボルを復号する。Ｓ２５２では非予測値を画素値３とする。Ｓ２５３の判定がＹｅｓならば、Ｓ２５４で予測値（ＭＰＳテーブル）７の反転・更新を行う。Ｓ２５５でＮＬＰＳテーブル１１参照による状態遷移を行う。また、Ｓ２５１の判定がＮｏならば優勢シンボルを復号する。Ｓ２５６では予測値７をそのまま画素値３とする。Ｓ２５７でＮＭＰＳテーブル１０参照による状態遷移を行う。

【００４１】図６０は、正規化処理を行うＲＥＮＯＲＭＤ処理を説明するフローチャートである。Ｓ２６１で変数ＣＴ５０が０か否か判定し、判定がＹｅｓならばＳ２６２でＢＹＴＥＩＮ処理を呼び出してＣレジスタ３０Ｂに１バイト符号４を入力する。Ｓ２６３ではＡレジスタ３１、Ｓ２６４ではＣレジスタ３０Ｂを１ビット上位へシフトすることによって２倍の乗算と等価な演算を行う。Ｓ２６５で変数ＣＴ５０から１を減じる。Ｓ２６６の判定は、正規化処理の終了判定を行っており、Ａレジスタ３１が０ｘ８０００未満ならばＳ２６１からＳ２６５を繰り返す。Ｓ２６７で変数ＣＴ５０値が０か否か判定し、判定がＹｅｓならばＳ２６８でＢＹＴＥＩＮ処理を呼び出してＣレジスタ３０Ｂに１バイト符号を入力する。

【００４２】図６１は、Ｃレジスタ３０Ｂに符号４を１バイトずつ読み込むＢＹＴＥＩＮ処理を説明するフローチャートである。ＳＣＤ（Stripe Coded Data）はストライプに対する符号４である。Ｓ２７１で判定がＹｅｓの場合、読み出す符号４がないため変数ＢＵＦＦＥＲ５１は０とする。Ｓ２７３で変数ＢＵＦＦＥＲ５１をＣＬＯＷレジスタ３６（Ｃｂ３７）に読み込み、Ｓ２７４で次の入力まで８ビットを処理するため変数ＣＴ５０に８を設定する。また、判定がＮｏの場合、１バイト符号４を変数ＢＵＦＦＥＲ５１に読み込む。

【００４３】図６２は、復号開始時のＳＴテーブル８、ＭＰＳテーブル７および各変数の初期値を設定するＩＮＩＴＤＥＣ処理を説明するフローチャートである。テーブルの初期値設定に関するＳ２８１、Ｓ２８２、Ｓ２９０については、符号化処理のＩＮＩＴＥＮＣ処理を説明するフローチャートのＳ１７１、Ｓ１７２、Ｓ１７７と同様である。Ｃレジスタ３０Ｂの初期値は、Ｃｘレジスタ３９およびＣｂレジスタ３７に符号４を３バイト読み込むことで設定される。Ｓ２８３でＣレジスタ３０Ｂをクリアし、Ｓ２８４でＢＹＴＥＩＮ処理を呼び出して符号４をＣｂレジスタ３７に１バイト読み込む。Ｓ２８５でＣレジスタ３０Ｂを８ビットシフトして、Ｓ２８６でＢＹＴＥＩＮ処理を呼び出して符号４をＣｂレジスタ３７に１バイト読み込む。Ｓ２８７でＣレジスタ３０Ｂを８ビットシフトして、Ｓ２８８でＢＹＴＥＩＮ処理を呼び出して符号４をＣｂレジスタ３７に１バイト読み込む。これで、合計３バイトの符号４をＣｘレジスタ３９およびＣｂレジスタ３７に設定したことになる。Ｓ２８９ではＡレジスタ３１の初期値を設定する。

【００４４】第２の従来例について

次に第２の従来例について説明する。第２の従来例による算術符号化の符号器・復号器は、日本国特許第２７５５０９１号に記載されるテーブルおよび処理を説明するフローチャートによって実現されるものである。ここで、符号をバイト単位に出力する算術符号化では、その出力バイト値が０ｘＦＦとなるとき、後に桁上がりが発生すると出力済みのより上位まで伝播する可能性があるため符号値を確定できない。そこで、出力バイト値が０ｘＦＦとなる時点で有効領域に桁上がりの可能性があれば強制的に桁上がり有無を確定させてしまうことを考える。

【００４５】図６３に強制的に桁上がり有無を確定させる概念図を示す。図において、下界値Ｃは符号レジスタ（Ｃレジスタ）値３０Ａであり、上界値Ｕ６０は下界値Ｃ３０Ａに有効領域幅Ａ（Ａレジスタ値）３１を加えた値となる。下界値および上界値の最上位ビットが一致しない場合に桁上がりありと判定され、そのとき有効領域内に桁上がり境界値Ｔ６１が存在することになる。

【００４６】よって、有効領域が桁上がり境界によって分割される２つの部分領域で、桁上がりがある領域を桁上がり領域、桁上がりがない領域を非桁上がり領域と呼ぶものとすれば、桁上がり領域幅Ｒ１６２は（Ｕ－Ｔ）、非桁上がり領域幅Ｒ０６３は（Ｔ－Ｃ）と表される。桁上がりの有無を確定するには、有効領域が桁上がり領域または非桁上がり領域に完全に一致するか、または含まれればよいことになり、ここでは強制的に有効領域をその部分領域の一方を切り捨て、他方を新たな有効領域として修正する。領域の切り捨てに当たり、より小さい領域を切り捨てた方が符号長のロスが少なくできるため、図中では、より小さいＲ０６３の部分領域を切り捨てている。部分領域を切り捨てることによって、修正

された有効領域に対する正規化処理の必要性を再度判定する必要がある。このような算術符号化の桁上がり制御方法を適応領域切捨て方式と呼ぶものとする。

【００４７】適応領域切捨て方式を適用した算術符号化の符号部１Ｂ、復号部１６Ｂの概略構成を図６４、図６５に示す。図６４では、算術符号器１３Ｂ、符号４Ｂの置換えに伴う符号化部１Ｂが第１の従来例の図４１に対応して構成される。図６５では、算術復号器１７Ｂ、符号４Ｂの置換えに伴う復号部１６Ｂが第１の従来例の図４２に対して異なる。これらを置き換えたブロック間のデータの流れ、また学習方法は従来の形態の図４１と図４２と同様とする。

【００４８】算術符号器１３Ｂは、ＬＳＺ値９と２値シンボル１４の入力から、Ｃレジスタ３０ＡとＡレジスタ３１で符号化演算が進められ、ＣＴカウンタ５０でバイト単位の符号出力タイミングを計り、符号出力時に同期して領域内に桁上がり境界の有無をＵレジスタ６０、Ｔレジスタ６１、Ｒ１レジスタ６２、Ｒ０レジスタ６３から判定し、桁上がり境界があれば適応領域切捨て方式を適用してＣレジスタ３０Ａ、Ａレジスタ３１を修正して桁上がりの有無を強制確定し、最大ＢＵＦＦＥＲ５１までの桁上がり伝搬を行って確定した符号出力を行い、符号４Ｂを出力する。

【００４９】また、算術復号器１７Ｂは、ＬＳＺ値９と符号４Ｂの入力から、ＣＴカウンタ５０でバイト単位の符号入力タイミングを計り、符号入力時に同期して領域内に桁上がり境界の有無を算術符号器１３Ｂの符号レジスタ３０Ａを再現したＤレジスタ６４、Ｕレジスタ６０、Ｔレジスタ６１、Ｒ１レジスタ６２、Ｒ０レジスタ６３から判定し、桁上がり境界があれば適応領域切捨て方式を適用してＣレジスタ３０Ａ、Ｄレジスタ６４、Ａレジスタ３１を修正して、ＢＵＦＦＥＲ５１を介して、Ｃレジスタ３０Ｂと、Ａレジスタ３１、Ｄレジスタ６４で復号演算が進められ、２値シンボル１４を出力する。

【００５０】図６４と図６５に示される構成ブロック内およびブロック間の詳細な処理動作は、説明済みの第１の従来例の処理を説明するフローチャートである図４６から図４９、図５３、図５４、図５６、図５７、図５９と、後述の処理を説明するフロー図の図６７から図７６で説明する。

【００５１】図６６に適応領域切捨て方式の符号レジスタを示す。上記第１の従来例に係る図４５に示した符号レジスタと比較してＣｓレジスタ３３が削除されている。これは、符号器のバイト出力と復号器のバイト入力の同期をとるためであり、適応領域切捨て処理の適用もバイト入出力と同期して適用するものとする。ここで、符号化（符号器）では、符号レジスタ値は有効領域の下界値、復号（復号器）では、符号レジスタ値は有効領域の下界値から符号値までの変位（オフセット）に更新されていくという違いがあるため、復号時には符号器の符号レジスタ値を再現していかないと桁上がり境界を正確に判断できない。よって、復号では、符号器の符号レジスタと同じ構成をとるＤレジスタ６４でその値を保持していくものとする。

【００５２】本第２の従来例において、修正あるいは追加された処理を説明するフローチャートについて説明する。適応領域切捨て方式を、上記第１の従来例で説明したＱＭ－Ｃｏｄｅｒの符号化／復号処理を説明するフローチャートに必要な修正のみ加えて説明を行う。符号器に関する図４６から図４９、図５３、図５４、復号器に関する図５６、図５７、図５９については、上記第１の従来例の処理を説明するフローと同様である。また、上記説明において使用した有効領域の上界値Ｕ６０、桁上がり境界値Ｔ６１、非桁上がり領域幅Ｒ０６３、桁上がり領域幅Ｒ１６２、Ｄレジスタ６４を変数として追加導入する。

【００５３】図６７は、正規化処理を行うＲＥＮＯＲＭＥ処理を説明するフローチャートである。上記第１の従来例（図５０）に対して、適応領域切捨て処理の適用を判定するためのＲＯＵＮＤＤＯＷＮＥ処理を呼び出すＳ１４７が追加され、BYTEOUT処理の呼び出し（図５０のＳ１４５）はＲＯＵＮＤＤＯＷＮＥ処理から行われる。

【００５４】図６８は、適応領域切捨て処理を適用するＲＯＵＮＤＤＯＷＮＥ処理を説明するフローチャートである。Ｓ３０１では出力バイト値が０ｘＦＦであるかを判定し、Ｎｏならばそのまま終了する。Ｓ３０２で桁上がり境界値Ｔ６１（定数）を設定し、Ｓ３０３で非桁上がり領域幅Ｒ０６３を設定する。Ｓ３０４で非桁上がり領域幅Ｒ０６３を領域幅（Ａレジスタ値）と比較し、Ｒ０値６３がＡレジスタ値３１より小さくない、またはＲ０値６３が０より大きくないならば（Ｎｏ）、桁上がり境界Ｔ６１が有効領域内にないためそのまま終了する。Ｒ０値６３がA値３１より小さい、またはＲ０値６３が０より大きいならば（Ｙｅｓ）、Ｓ３０５で上界値Ｕ６０を設定し、Ｓ３０６で桁上がり領域幅Ｒ１６２を設定する。Ｓ３０７でＲ０値６３がＲ１値６２より小さければ（Ｙｅｓ）、桁上がり領域を有効領域にするため、Ｓ３０８で下界値を桁上がり境界値に変更し、Ｓ３０９で領域幅（Ａレジスタ値）にＲ１値を設定し、大きければ（Ｎｏ）、非桁上がり領域を有効領域にするためＳ３０９で領域幅（Ａレジスタ値）３１にＲ０値６３を設定する。最後にＳ３１１でＢＹＴＥＯＵＴ処理を呼び出す。

【００５５】図６９（ＢＹＴＥＯＵＴ処理を説明するフロー）、図７０（ＩＮＩＴＥＮＣ処理を説明するフロー）及び図７１（ＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフロー）の図５１、図５２及び図５５に対するそれぞれの変更を説明すると、まず、符号レジスタのＣｓレジスタ部３３の削除によって、図６９（ＢＹＴＥＯＵＴ処理を説明するフロー）のＳ１５１'、Ｓ１５７'、図７０（ＩＮＩＴＥＮＣ処理を説明するフロー）のＳ１７

６'、図７１（ＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフロー）のＳ２０２'、Ｓ２０５'、Ｓ２０６'のマスクやシフト数の設定が変更される。

【００５６】また、図６８（ＲＯＵＮＤＤＯＷＮＥ処理を説明するフロー）において、出力バイト値が０ｘＦＦであるときの桁上がり伝搬の有無についてすでに解決済みであるため、変数ＢＵＦＦＥＲを逐次出力していくことができるようになり、図６９（ＢＹＴＥＯＵＴ処理を説明するフロー）の判定Ｓ１５９は不要となり、同時に変数ＳＣ自体ととそれに関わる図６９（ＢＹＴＥＯＵＴ処理を説明するフロー）のＳ１５４、Ｓ１５５、Ｓ１６０、Ｓ１６２、Ｓ１６３、図７０（ＩＮＩＴＥＮＣ処理を説明するフロー）のＳ１７３、図７１（ＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフロー）のＳ２０４、Ｓ２０８も不要となる。

【００５７】図７２は、上方領域を復号するＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。

上記第１の従来例（図５８）に対して、符号器の符号レジスタをＤレジスタ６４として再現するためのＳ２４２、Ｓ２４３が追加されている。

【００５８】また、図７３は、正規化処理を行うＲＥＮＯＲＭＤ処理を説明するフローチャートである。上記第１の従来例（図６０）に対して、符号器の符号レジスタをＤレジスタとして再現するためのＳ２６９、適応領域切捨て処理の適用を判定するためのＲＯＵＮＤＤＯＷＮＤ処理を呼び出すＳ２７０が追加され、BYTEOUT処理の呼び出し（図６０のＳ２６２）はＲＯＵＮＤＤＯＷＮＤ処理から行われる。また、図６０のＳ２６１とＳ２６７の判定の冗長性を除去したため処理を説明するフロー形状が異なるが、処理内容は同様である。

【００５９】図７４は、適応領域切捨て処理を適用するＲＯＵＮＤＤＯＷＮＤ処理を説明するフローチャートである。図６８のＲＯＵＮＤＤＯＷＮＥ処理に対して、符号レジスタ変数名Ｃ３０ＡがＤ６４に変更されており、Ｓ３２１からＳ３３１はＳ３０１からＳ３１１に対応し、復号器の符号レジスタ３０Ｂに符号器の符号レジスタ３０Ａ（レジスタＤ６４）に対応する適応領域切捨て処理を行うためのＳ３３２が追加されている。また、最後に図６８のＢＹＴＥＯＵＴ処理（Ｓ３１１）の代わりにＳ３３１でＢＹＴＥＩＮ処理ををを呼び出す。

【００６０】図７５（ＢＹＴＥＩＮ処理を説明するフロー）、図７６（ＩＮＩＴＤＥＣ処理を説明するフロー）の図６１、図６２に対するそれぞれの変更として、符号器の符号レジスタ３０ＡをＤレジスタ６４として再現するための初期化処理である図７５のＳ２７６、図７６のＳ２９０が追加されている。

【００６１】

【発明が解決しようとする課題】ところで、上述した従来例では次のような問題点がある。第１の従来例に係る符号器では、桁上がりの有無が解決せずに符号値が確定できない場合にカウンタで待機する符号長を記憶させるが、データ長の制限がないため桁あふれする可能性があるという課題があった。また、符号値が確定しなければ最後まで符号が掃き出されないこともあり、特にデータ長が不明な場合には符号の確定遅延時間を見積もれないという課題があった。さらに符号値が確定したとき、出力を待機していた符号長が長ければその掃き出し時間も長くなり、符号化処理を全くまたは一時的に停止させなければならないという課題があった。

【００６２】他方、第２の従来例に係る符号器では、符号レジスタの値により桁上がり制御の実行を判定し、桁上がり境界を厳密に計算しなければならないという課題があった。また、第２の従来例に係る復号器では、符号器の符号レジスタを再現しなければ桁上がり制御の正確な実行判定ができないという課題があった。

【００６３】この発明は、以上のような従来例に係る課題を解決するためになされたものであり、符号が確定するまで待機する符号長を記憶するカウンタの桁あふれをなくし、適当な間隔で符号を強制的に確定することにより桁上がりを制御して符号の確定遅延時間を有限化して見積もれるようにすると共に、符号値が確定したときに所定のパターンへの置き換えによって総符号長と符号の掃出し時間を短縮することを目的とする。

【００６４】また、桁上がり境界を厳密に計算せずに桁上がり制御を行い、復号器で符号器の符号レジスタを再現せずに桁上がり制御を行うことを目的とする。

【００６５】

【課題を解決するための手段】この発明に係る符号化装置は、情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータメモリと、上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される符号化パラメータを出力する確率推定テーブルと、上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号器とを備える符号化装置において、所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、上記符号器は、上記桁上がり検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００６６】また、上記符号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記桁上がり境界検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００６７】また、上記桁上がり境界検出器は、検出された桁上がり境界を含む部分領域を切り捨てるように指示することを特徴とするものである。

【００６８】また、上記所定の間隔は、所定単位の符号化対象データを符号化したときであることを特徴とするものである。

【００６９】また、上記所定の間隔は、符号値の確定まで待機させる符号長の最大値を基準に設定されることを特徴とするものである。

【００７０】また、上記符号器は、上記所定の間隔を基準に符号値の確定まで待機させる符号長の最大値が設定されることを特徴とするものである。

【００７１】また、上記所定の間隔は、所定単位の符号出力が発生したときであることを特徴とするものである。

【００７２】また、上記所定の間隔は、所定単位の符号出力が発生し、かつ符号出力値が所定値となるときであることを特徴とするものである。

【００７３】また、上記桁上がり境界検出器は、上記有効領域内に桁上がり境界が検出されなくても上記有効領域を上位と下位に等分した部分領域のどちらか一方の部分領域を指示することを特徴とするものである。

【００７４】また、上記符号器は、確定した符号に同じ値が連続するときにその値と長さを特定できるランレングスマーカに置き換えるランレングスマーカ変換器を有することを特徴とするものである。

【００７５】また、上記ランレングスマーカ変換器は、置き換えずにそのまま出力した方が変換される上記ランレングスマーカより符号出力が短いときには変換を行わないことを特徴とするものである。

【００７６】また、上記ランレングスマーカから抽出される連続長の最大値は、復号の際のランレングスマーカ逆変換の結果に基づいて上記符号器が上記ランレングスマーカ変換器に設定することを特徴とするものである。

【００７７】また、この発明に係る復号化装置は、復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータメモリと、上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される復号パラメータを出力する確率推定テーブルと、上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号器とを備える復号化装置において、所定の単位の情報源データの出力処理または符号の入力を所定の間隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、上記復号器は、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００７８】また、上記復号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００７９】また、上記桁上がり境界検出器は、検出された桁上がり境界を含む部分領域を切り捨てるように指示することを特徴とするものである。

【００８０】また、上記所定の間隔は、所定単位の復号対象データを復号したときであることを特徴とするものである。

【００８１】また、上記所定の間隔は、所定単位の符号入力が発生したときであることを特徴とするものである。

【００８２】また、上記所定の間隔は、所定単位の符号入力が発生し、かつ復号と同時に再現される符号出力値が所定値となるときであることを特徴とするものである。

【００８３】また、上記復号器は、ランレングスマーカを検出すると、置き換えられた長さの元の符号値に逆変換するランレングスマーカ逆変換器を有することを特徴とするものである。

【００８４】また、この発明に係る符号化復号化装置は、符号化装置として、情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータメモリと、上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される符号化パラメータを出力する確率推定テーブルと、上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号器と、所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、上記符号器は、上記桁上がり検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新すると共に、復号化装置として、復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータメモリと、上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習メモリと、上記学習データを基に指定される復号パラメータを出力する確率推定テーブルと、上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号器と、所定の単位の情報源データの出力処理または符号の出力を所定の間

隔として計測して通知する同期検出器と、上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出器とを備え、上記復号器は、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００８５】また、上記符号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記桁上がり境界検出器により指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新すると共に、上記復号器は、上記桁上がり境界検出器で桁上がり境界が検出されたとき、上記有効領域を上位と下位に等分した部分領域の上記符号値を含まない一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００８６】また、この発明に係る符号化方法は、

（ａ）情報源データを蓄積して符号化対象データとその補助的なパラメータ（コンテクスト）を出力するデータ蓄積ステップと、（ｂ）上記補助パラメータを基に指定される上記符号化対象データに関する学習データを蓄積し出力する学習ステップと、（ｃ）上記学習データを基に指定される符号化パラメータを出力する確率推定ステップと、（ｄ）上記符号化対象データと上記符号化パラメータを基に算術符号化を行い符号を出力する符号化ステップと、（ｅ）所定の単位の情報源データの入力処理または符号の出力を所定の間隔として計測して通知する同期検出ステップと、（ｆ）上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出ステップと、（ｇ）上記桁上がり境界検出ステップにより指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新する符号化修正ステップとを有するものである。

【００８７】また、上記符号化修正ステップは、上記桁上がり境界検出ステップで桁上がり境界が検出されたとき、上記桁上がり境界検出ステップで指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００８８】また、上記符号化ステップおよび上記符号化修正ステップは、確定した符号に同じ値が連続するときにその値と長さを特定できるランレングスマーカに置き換える変換ステップを有することを特徴とするものである。

【００８９】上記変換ステップは、置き換えずにそのまま出力した方が変換される上記ランレングスマーカより符号出力が短いときには変換を行わないことを特徴とするものである。

【００９０】また、上記ランレングスマーカから抽出される連続長の最大値は、復号の際、ランレングスマーカ逆変換の結果に基づいて上記符号化ステップが上記変換ステップに設定することを特徴とするものである。

【００９１】また、この発明に係る復号化方法は、

（ａ）復号対象データに対する補助的なパラメータ（コンテクスト）を出力し復号された上記復号対象データを蓄積して情報源データとして出力するデータ蓄積ステップと、（ｂ）上記補助パラメータを基に指定される上記復号対象データに関する学習データを蓄積し出力する学習ステップと、（ｃ）上記学習データを基に指定される復号パラメータを出力する確率推定ステップと、（ｄ）上記復号パラメータと符号を基に算術復号を行い上記復号対象データを出力する復号化ステップと、（ｅ）所定の単位の情報源データの出力処理または符号の入力を所定の間隔として計測して通知する同期検出ステップと、（ｆ）上記所定の間隔で有効領域内の桁上がり境界値を検出すると共に、その検出結果に基づいて有効領域内の切り捨て領域を指示する桁上がり境界検出ステップと、（ｇ）上記桁上がり境界検出ステップにより指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新する復号化修正ステップとを有するものである。

【００９２】また、上記復号化修正ステップは、上記桁上がり境界検出ステップで桁上がり境界が検出されたとき、上記桁上がり境界検出ステップで指示された上記有効領域を上位と下位に等分した部分領域の一方を切り捨てて上記有効領域を更新することを特徴とするものである。

【００９３】さらに、上記復号化ステップは、ランレングスマーカを検出すると、置き換えられた長さの元の符号値に逆変換するランレングスマーカ逆変換ステップを有することを特徴とするものである。

【００９４】

【発明の実施の形態】実施の形態１．本実施の形態１では、上述した第２の従来例と同様に、有効領域の部分領域を切り捨て、出力済みの符号に対して桁上がり有無を確定する方法をとるが、有効領域を２つの部分領域に等分割し、桁上がり境界を含む部分領域を切り捨てるものである。

【００９５】領域幅は常に符号器と復号器とで同値をとるため、等分割操作によれば、上記第２の従来例のように桁上がり境界値を厳密に符号化／復号に持ち込まなくても上位／下位のどちらの部分領域に含むかだけわかればよいことになる。

【００９６】例えば、図１に示すように、最大有効領域を１６とするとき、正規化処理により有効領域は８以上に保たれる。このとき、１．０００００を桁上がり境界とすれば、その境界が上位にあれば上位１／２領域を切り捨て、下位にあれば下位１／２領域を切り捨てる。ただし、有効領域幅が奇数のとき、有効な表現精度の不足に

よる桁落ちによる誤差が生じるため、その解決策も含める必要がある。同図では、有効領域幅＝９であるため、単純に１／２とすれば４（整数）となり、それを下位領域幅とすると上位領域幅は５となる。よって、上位領域を切り捨てれば下位領域幅が２倍に拡大されて８、下位領域を切り捨てれば上位領域幅が２倍に拡大されて１０となり、厳密に等しく分割されたことにならないことになる。

【００９７】ここで、領域幅（Ａレジスタ）３１の１／２を計算した後、正規化処理で少なくとも２倍に拡大されることは確実である。よって、Ａレジスタの１／２演算で桁落ちするビットを後から補正する方法がある。また、もう１つの方法は、領域幅（＝９）は保ったまま、まず下界値（Ｃレジスタ）３０Ａだけを２倍し、下位領域を切り捨てるときその領域幅３１＝９を加えれば所望の誤差のない座標を得ることができる。いずれにしても、１／２領域切捨て方式によれば、１回適用につき必ず１ビットの符号長が発生することになる。

【００９８】本実施の形態１では、１／２領域切捨て方式の算術符号化の符号器を符号部１Ｃ、その復号器を復号部１６Ｃとし、その符号器、復号器の概略構成を図２、図３に示す。図２では、算術符号器１３Ｃ、符号４Ｃの置換えに伴う符号化部１Ｃが第２の従来例の図６４に対応して構成される。図３では、算術復号器１７Ｃ、符号４Ｃの置換えに伴う復号部１６Ｃが第２の従来例の図６５に対して異なる。これらを置き換えたブロック間のデータの流れ、また学習方法は従来例の図６４と図６５と同様とする。

【００９９】算術符号器１３Ｃは、ＬＳＺ値９と２値シンボル１４の入力から、Ｃレジスタ３０ＡとＡレジスタ３１で符号化演算が進められ、ＣＴカウンタ５０でバイト単位の符号出力タイミングを計り、符号出力時に同期して領域内に桁上がり境界の有無をＴレジスタ６１、Ｒ０レジスタ６３から判定し、桁上がり境界があれば１／２領域切捨て方式を適用してＣレジスタ３０Ａ、Ａレジスタ３１を修正して桁上がりの有無を強制確定し、最大ＢＵＦＦＥＲ５１までの桁上がり伝搬を行って確定した符号出力を行い、符号４Ｃを出力する。

【０１００】また、算術復号器１７Ｃは、ＬＳＺ値９と符号４Ｃの入力から、ＣＴカウンタ５０でバイト単位の符号入力タイミングを計り、符号入力時に同期して領域内に桁上がり境界の有無を算術符号器１３Ｃの符号レジスタ３０Ａを再現したＤレジスタ６４、Ｔレジスタ６１、Ｒ０レジスタ６３から判定し、桁上がり境界があれば１／２領域切捨て方式を適用してＣレジスタ３０Ａ、Ｄレジスタ６４、Ａレジスタ３１を修正して、ＢＵＦＦＥＲ５１を介して、Ｃレジスタ３０Ｂと、Ａレジスタ３１、Ｄレジスタ６４で復号演算が進められ、２値シンボル１４を出力する。

【０１０１】図２と図３に示される構成ブロック内およびブロック間の詳細な処理動作は、説明済みの第１の従来例の処理を説明するフロー図である図４６から図４９、図５３、図５４、図５６、図５７、図５９と、同じく第２の従来例の処理を説明するフローチャートである図６７、図６９から図７４、図７５、図７６と、後述の処理を説明するフロー図４から図７、またはその等価な処理を説明するフローとして与えられる図４、図６、図８から図１１で説明する。

【０１０２】本実施の形態１では、第２の従来例において桁上がりまたは非桁上がり領域領域の切捨て処理を行っている図６８（ＲＯＵＮＤＤＯＷＮＥ処理を説明するフロー）、図７４（ＲＯＵＮＤＤＯＷＮＤ処理を説明するフロー）を次に示す図４、図６へ変更し、そこから呼び出される図５（ＨＡＬＶＩＮＧＥ処理を説明するフロー）、図７（ＨＡＬＶＩＮＧＤ処理を説明するフロー）を追加することによって１／２領域切捨て方式を実現する方法を示す。

【０１０３】図４は、１／２領域切捨て処理の適用を判定するＲＯＵＮＤＤＯＷＮＥ処理を説明するフローチャートである。図中、Ｓ３０１からＳ３０４、およびＳ３０１、Ｓ３０４の判定結果（いずれもＮｏ）により処理されるＳ３１１は図６８と同じである。Ｓ３０４の判定結果（Ｙｅｓ）によりＳ３１２で１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理を呼び出す。

【０１０４】図５は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。Ｓ３４１で非桁上がり領域幅Ｒ０の２倍がＡレジスタ値３１より小さければ（判定Ｙｅｓ）、下位１／２領域が桁上がり境界を含み、小さくなければ（判定Ｎｏ）、情報１／２領域が桁上がり領域を含むことになり、桁上がり境界を含む１／２領域を切り捨てるものとする。下位１／２領域を切り捨てる場合、Ｓ３４２でＣレジスタ値（有効領域下界値）３０ＡをＡレジスタ値（有効領域幅）３１の１／２だけ更新した後、Ｓ３４３でＢＹＴＥＯＵＴ処理を呼び出す。領域切捨てに伴い、実際はＡレジスタ値３１は１／２とするべきであるが、すぐに正規化処理で２倍に拡大されるため更新していない。Ｃレジスタ３０ＡとＣＴカウンタ値５０に対して正規化処理に相当するＳ３４４、Ｓ３４５を行う。Ｓ３４６でＡレジスタ値３１が奇数か否かを判定し、奇数であればＡレジスタ値３１の１／２倍で桁落ちした分をＳ３４７で補正する。上位１／２領域を切り捨てる場合、Ｃレジスタ値３０Ａは更新されず、Ｓ３４８でＢＹＴＥＯＵＴ処理を呼び出す。Ｃレジスタ３０ＡとＣＴカウンタ値５０に対して正規化処理に相当するＳ３４９、Ｓ３５０を行う。

【０１０５】図６は、１／２領域切捨て処理の適用を判定するＲＯＵＮＤＤＯＷＮＤ処理を説明するフローチャートである。図中、Ｓ３２１からＳ３２４、およびＳ３２１、Ｓ３２４の判定結果（いずれもＮｏ）により処理されるＳ３３２は図７４と同じである。Ｓ３２４の判定

結果（Ｙｅｓ）によりＳ３３３で１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理を呼び出す。

【０１０６】図７は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。図５のＲＯＵＮＤＤＯＷＮＥ処理に対して、符号レジスタ変数名Ｃ３０ＡがＤ６４に変更されており、Ｓ３６１からＳ３７０はＳ３４１からＳ３５０に対応し、復号器の符号レジスタＣ３０Ｂに符号器の符号レジスタ３０Ａ（レジスタＤ６４）に対応する１／２領域切捨て処理を行うためのＳ３７１、Ｓ３７２、Ｓ３７３、Ｓ３７４を追加している。また、最後に図６８のＢＹＴＥＯＵＴ処理（Ｓ３１１）の代わりにＳ３３１でＢＹＴＥＩＮ処理を呼び出す。

【０１０７】ここで、図５のＨＡＬＶＩＮＧＥ処理を説明するフロー、図７のＨＡＬＶＩＮＧＤ処理を説明するフローは、呼び出されるＢＹＴＥＯＵＴ処理を説明するフロー（図６９）、ＢＹＴＥＩＮ処理を説明するフロー（図７４）とともに、次に示す図８（ＨＡＬＶＩＮＧＥ処理を説明するフロー）、図９（ＢＹＴＥＯＵＴ７処理を説明するフロー）、図１０（ＨＡＬＶＩＮＧＤ処理を説明するフロー）、図１１（ＢＹＴＥＩＮ７処理を説明するフロー）に置き換えることができる。ＢＹＴＥＯＵＴ７処理を説明するフロー、ＢＹＴＥＩＮ７処理を説明するフローはＡレジスタ値（領域幅）３１を１／２にしないでＣレジスタ３０Ｂを先に２倍に拡大して桁落ちを防止するため、バイトの入出力位置がＢＹＴＥＯＵＴ処理を説明するフロー、ＢＹＴＥＩＮ処理を説明するフローチャートの正規の位置より上位に１ビットだけずれている。特に、符号器では上位に１ビット余分に精度が保証されていなければならない。

【０１０８】図８は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。Ｓ３８１、Ｓ３８２でＣレジスタ３０Ａ、非桁上がり領域幅Ｒ０６３を２倍に拡大し、正規化に相当する処理を先に行う。Ｓ３０７で非桁上がり領域幅Ｒ０６３がＡレジスタ値３１より小さければ（判定Ｙｅｓ）、下位１／２領域が桁上がり境界を含み、小さくなければ（判定Ｎｏ）、上位１／２領域が桁上がり領域を含むことになる。下位１／２領域を切り捨てる場合（判定Ｙｅｓ）、Ｓ３８４でＣレジスタ値（有効領域下界値）３０ＡをＡレジスタ値（有効領域幅）３１だけ更新する。符号バイト出力前に本来はＢＹＴＥＯＵＴ処理の直後に行われる２倍の拡大（正規化処理）を先行しているため、Ｓ３８５では、ＢＹＴＥＯＵＴ処理ではなく、ＢＹＴＥＯＵＴ７処理を呼び出す。

【０１０９】図９は、ＢＹＴＥＯＵＴ処理の詳細な処理を説明するフローチャートである。このＢＹＴＥＯＵＴ７処理の図６９（ＢＹＴＥＯＵＴ処理を説明するフロー）に対する変更を示す。Ｓ１５１"、Ｓ１５７"の修正は、符号出力位置が１ビット上位に移動していることによる。Ｓ１５８'の修正は、図８ですでに正規化に相当する処理（Ｓ３８２）が行われていることによる。

【０１１０】図１０は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。

図８のＲＯＵＮＤＤＯＷＮＥ処理に対して、符号レジスタ変数名Ｃ３０ＡがＤ６４に変更されており、Ｓ３９１からＳ３９５はＳ３８１からＳ３８５に対応し、復号器の符号レジスタＣ３０Ｂに符号器の符号レジスタ３０Ａ（レジスタＤ６４）に対応する処理を行うためのＳ３９６、Ｓ３９７を追加している。また、最後に図８のＢＹＴＥＯＵＴ７処理（Ｓ３８５）の代わりにＳ３９５でＢＹＴＥＩＮ７処理を呼び出す。

【０１１１】図１１は、ＢＹＴＥＩＮ７処理の詳細な処理を説明するフローチャートである。このＢＹＴＥＩＮ７処理の図７４（ＢＹＴＥＩＮ処理を説明するフロー）に対する変更を示す。Ｓ２７３'、Ｓ２７６'の修正は、符号入力位置が１ビット上位に移動していることによる。Ｓ２７４'の修正は、図１０ですでに正規化に相当する処理（Ｓ３９２、Ｓ３９６）が行われていることによる。

【０１１２】上述した実施の形態１によれば、桁上がり境界が上位と下位の１／２領域のどちらに存在しているかのみ判定されればよいため、桁上がり領域幅Ｒ１６２を計算する必要がない。

【０１１３】実施の形態２．前述した第１の従来例に示されるＱＭ－Ｃｏｄｅｒでは、出力済みであっても確定していない符号を変数ＢＵＦＦＥＲ５１、ＳＣカウンタ５２で待機させているが、符号器の仕様によっては符号化中にＳＣカウンタ５２の桁あふれが発生することがあり、結果として桁あふれ１回ごとにそのカウント最大数だけ符号バイト（０ｘＦＦまたは０ｘ００）が欠落し、正確な符号を出力できないことになる。

【０１１４】一般に、最終的な画像サイズは事前にわかっているとは限らないが、有限のサイズで区切ったストライプ（またはライン）単位で処理を行えばその最悪の符号量を見積もることができる。算術符号化では最悪の場合１ビットを若干超える程度となることが知られているため、その区切ったストライプ（またはライン）構成サイズ（バイト数）に対してほぼ用意すべきＳＣカウンタ５２の仕様が確定でき、その２倍の符号バイト０ｘＦＦのランを計数できれば十分である。本実施の形態２では、ストライプ終了ごとにそこまで出力を待機している符号バイトを強制的に確定して掃き出していくことによって、最大確定遅延もストライプの符号化処理時間程度に見積もれるようになる。この符号バイトの確定手法として１／２領域切捨て方式を適用するものとする。

【０１１５】また、前述した第２の従来例に示される適応領域切捨て方式では、１バイトの符号バイトのみ待機させれば出力を逐次行っていくことが可能であるが、領域切捨て処理による最小の符号長ロスで抑えるために有

効領域内の桁上がり境界の位置を厳密に計算しなければならないこと、また、そのため、復号器では符号器の符号レジスタ３０Ａの再現を行わなければ正確な復号ができない。さらに、適応領域切捨て処理の実行をＣｓレジスタ３３の削除によって符号器、復号器のバイト入出力のタイミングで同期させなければ他の専用同期手段を用意する必要があり、その判定などによって符号化／復号の負荷が増加する。例えば、Ｃｓレジスタ３３を削除せずに符号器と復号器を同期させるために、符号器には上記専用同期手段として正規化シフト数のカウンタを設け、８ビットごと（８の倍数時）に同期タイミングとみなすか、または復号のＣＴカウンタ５０を再現すればよい。

【０１１６】本実施の形態２では、ストライプの終了のタイミングで１／２領域切捨て処理を行うことによってＣｓレジスタ３３の有無による同期上の制約を解消できる。また、Ｃレジスタ３０Ａから出力される符号バイト値に依存せず、かつ有効領域内に桁上がり境界がないときにも必ず１／２領域切捨て処理を行うことによって、復号器では符号器の符号レジスタ３０Ａの再現することなく、また、桁上がり境界を厳密に計算せずに符号（Ｃレジスタ値）３０Ｂが指し示さない上位または下位の１／２領域を切り捨てることによって正確な復号を行うことができる。

【０１１７】ここで、符号器は、上記のように切り捨てるべき１／２領域を決定し、有効領域を更新するが、これはダミーシンボルの符号化とみなせる。また、復号器は、符号の指し示す１／２領域に有効領域を更新するが、これはダミーシンボルの復号とみなすことができ、有効領域が上位／下位のどちらに更新されてもシンボルは無視してよい。

【０１１８】本実施の形態２の１／２領域切捨て方式では、ストライプの符号化処理で符号ビットが発生しなくても１／２領域切捨てによって１ビットの符号が発生する。このため、ＳＣカウンタ５２の桁あふれを抑えながら、最悪８ストライプで１バイトの符号を出力させられることになり、その所要時間を符号確定までの最大遅延として見積もることができる。

【０１１９】１／２領域切捨て方式の算術符号化の符号器を符号部１Ｃ、その復号器を復号部１６Ｃとし、その符号器、復号器の概略構成を、図１２、図１３に示す。図１２では、算術符号器１３Ｄ、符号４Ｄの置換えに伴う符号化部１Ｄが第１の従来例の図３９に対応して構成される。図１３では、算術復号器１７Ｄ、符号４Ｄの置換えに伴う復号部１６Ｄが第１の従来例の図４０に対して異なる。これらを置き換えたブロック間のデータの流れ、また、学習方法は第１の従来例の図３９と図４０と同様とする。

【０１２０】算術符号器１３Ｄは、ＬＳＺ値９と２値シンボル１４の入力から、Ｃレジスタ３０ＡとＡレジスタ３１で符号化演算が進められ、ＣＴカウンタ５０でバイト単位の符号出力タイミングを計り、第１の従来例と同様にＢＵＦＦＥＲ５１とＳＣカウンタで桁上がりの伝搬可能性のある未確定の符号出力待機と確定した符号出力を行い、符号４Ｄを出力していく。そして、ライン終端（あるいはストライプ終端）の検出に同期して領域内に桁上がり境界の有無をＴレジスタ６５、ＣＷレジスタ６６から判定し、選択された上位または下位の１／２切捨て領域を切り捨てる１／２領域切捨て方式を適用し、Ｃレジスタ３０Ａ、Ａレジスタ３１を修正して桁上がりの有無を強制確定し、最大ＢＵＦＦＥＲ５１までの桁上がり伝搬を行って確定した符号出力を行い、符号４Ｄを出力する。

【０１２１】また、算術復号器１７Ｄは、ＬＳＺ値９と符号４Ｄの入力から、ＣＴカウンタ５０でバイト単位の符号入力タイミングを計り、ＢＵＦＦＥＲ５１を介して、Ｃレジスタ３０Ｂと、Ａレジスタ３１で復号演算が進められ、２値シンボル１４を出力する。そして、ライン終端（あるいはストライプ終端）の検出に同期してＣレジスタ３０Ｂの指し示さない側の１／２切捨て領域を切り捨てる１／２領域切捨て方式を適用し、Ｃレジスタ３０Ｂ、Ａレジスタ３１を修正する。

【０１２２】図１２と図１３に示される構成ブロック内およびブロック間の詳細な処理動作は、説明済みの第１の従来例の処理を説明するフローチャートである図４７から図５５、図５７から図６２と、後述の処理を説明するフロー図１４から図１９で説明する。本実施の形態２では、図１４および図１８により、ストライプの終了時に１／２領域切捨て方式を適用する場合を示す。

【０１２３】本実施の形態２における１／２領域切捨て方式の一例を、上記第１の従来例で説明したＱＭ－Ｃｏｄｅｒの符号化／復号処理を説明するフローチャートに必要な修正のみ加えて説明を行う。符号化／復号／領域幅レジスタに関する図４５、符号器に関する図４７から図５５、復号器に関する図５７から図６２については、上記第１の従来例の処理を説明するフローと同様である。また、説明上、上記第１の従来例に追加導入される変数は、有効領域の１／２領域境界値Ｔ６５、変数ＣＴ値５０によって定義される桁上がり境界値ＣＷ６６である。

【０１２４】図１４は、符号化処理の全体の流れを示すＥＮＣＯＤＥＲ処理を説明するフローチャートである。

図４６におけるストライプの終了判定（Ｓ１０４）と画像の終了判定（Ｓ１０５）の間に１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理の呼出し（Ｓ１０７）を追加する。

【０１２５】図１５は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。Ｓ５０１、Ｓ５０２は、Ｃレジスタ３０Ａ、ＣＴカウンタ５０に対する正規化処理に相当し、Ａレジスタ３１は

そのまま保つ。Ｓ５０３、Ｓ５０４は、有効領域の１／２領域境界値Ｔ６５、ＣＴカウンタ値５０で定義される桁上がり境界値ＣＷ６６を設定する。Ｓ５０５で、１／２領域境界値Ｔ６５と桁上がり境界値ＣＷ６６を比較判定する。判定結果Ｙｅｓならば桁上がり境界は上位１／２領域以上に存在するため、桁上がり伝搬がない待機中の符号バイトを出力し、下位１／２領域を有効領域とするＣＯＤＥＬＯＷＥＲ処理を呼び出す（Ｓ５０６）。判定結果Ｎｏならば桁上がり境界は下位１／２領域以下に存在するため、桁上がり伝搬させた待機中の符号バイトを出力し、上位１／２領域を有効領域とするＣＯＤＥＵＰＰＥＲ処理を呼び出す（Ｓ５０７）。最後に、Ｓ５０８で変数ＣＴ５０が０ならば、Ｓ５０９でＢＵＦＦＥＲ５１を書き出し、Ｓ５１０でＣｂレジスタ値３４をＢＵＦＦＥＲ５１に設定し、Ｓ５１１でＣｃレジスタ部３５およびＣｂレジスタ部３４をクリアし、Ｓ５１２でＣＴカウンタ５０に８を設定する。ここで、すでにＳ５０６あるいはＳ５０７によって、桁上がり有無を確定し、桁上がりがあれば伝搬処理もされ（伝搬後、Ｃｃレジスタ部３５＝０）、待機していた符号バイトの掃出しが完了（掃出し後、変数ＳＣ値５２＝０）しており、Ｓ５０９からＳ５１２はＢＹＴＥＯＵＴ処理（図５１）において冗長部を最大限省いたＳ１６１、Ｓ１６４（ＴＥＭＰ＝Ｃｂレジスタ値）、Ｓ１５７、Ｓ１５８に相当する。Ｓ５１０ではＢＵＦＦＥＲ５１がバイト０ｘＦＦになっても問題ない。

【０１２６】図１６は、上位１／２領域を切り捨て、桁上がりなしの出力待機符号の掃出し処理を行うＣＯＤＥＬＯＷＥＲ処理を説明するフローチャートである。Ｓ５２１で変数ＳＣ値５２が正ならば、Ｓ５２２でＢＵＦＦＥＲ５１を書き出す。さらに、Ｓ５２３で変数ＳＣ値５２が１より大きければ、Ｓ５２４で符号バイト０ｘＦＦを（ＳＣ-１）回書き出す。Ｓ５２５で最後の待機符号バイトである０ｘＦＦをＢＵＦＦＥＲ５１に設定し、Ｓ５２６で変数ＳＣ値５２を０とする。

【０１２７】図１７は、下位１／２領域を切り捨て、桁上がりありの出力待機符号の掃出し処理を行うＣＯＤＥＵＰＰＥＲ処理を説明するフローチャートである。Ｓ５３１でＢＵＦＦＥＲ５１に桁上がりを伝搬させ、Ｃレジスタ３０Ａから桁上がり分（桁上がり境界値ＣＷ６６）をクリアする。Ｓ５３３で変数ＳＣ値５２が正ならば、Ｓ５３４でＢＵＦＦＥＲ５１を書き出す。さらに、Ｓ５３５で変数ＳＣ値５２が１より大きければ、Ｓ５３６で符号バイト０ｘ００を（ＳＣ-１）回書き出す。Ｓ５３７で最後の待機符号バイトである０ｘ００をＢＵＦＦＥＲ５１に設定し、Ｓ５３７で変数ＳＣ値５２を０とする。

【０１２８】図１８は、復号処理の全体の流れを示すＤＥＣＯＤＥＲ処理を説明するフローチャートである。図５６におけるストライプの終了判定（Ｓ２１４）と画像の終了判定（Ｓ２１５）の間に１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理の呼出し（Ｓ２１６）を追加する。

【０１２９】図１９は、１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。

Ｓ５４１、Ｓ５４２は、Ｃレジスタ３０Ｂ、ＣＴカウンタ５０に対する正規化処理に相当し、Ａレジスタ３１はそのまま保つ。Ｓ５４５でＣＨＩＧＨレジスタ値３８とＡレジスタ値３１を比較し、判定結果Ｙｅｓであれば下位１／２領域が切り捨てられたことになってＳ５４６でＣＨＩＧＨレジスタ値３８からＡレジスタ値３１を減じる。判定結果Ｎｏであれば下位１／２領域が切り捨てられたことになる。最後に、Ｓ５４７で変数ＣＴ５０が０ならば、ＢＹＴＥＩＮ処理を呼び出す。

【０１３０】ただし、桁上がり境界値ＣＷ６６が１／２領域境界Ｔ６５と等しい場合は、上位、下位のどちらの１／２部分領域をとってもよいが、例えば国際標準勧告Ｔ．８２では符号バイト０ｘＦＦの直後にはＳＴＵＦＦバイト（０ｘ００）を挿入して伝送上の制御（マーカコードの確保）を行うため、桁上がりを伝搬させた方が最終的な符号長が短くなることがある。

【０１３１】ここでは、画像を区切ったストライプ（またはライン）構成サイズ（バイト数）に対して用意すべきＳＣカウンタ５２の仕様が確定できるとしたが、逆にＳＣカウンタ５２の仕様が変更できないときには桁あふれしえないストライプ（またはライン）構成サイズを設定すればよい。

【０１３２】実施の形態３．上記実施の形態２の最大遅延が８ストライプであったが、符号ビットが発生しないときは桁上がりの有無は解決されたままになっているため、１／２領域切捨て処理を行わなければ符号確定までの遅延は大きくなるが、ＳＣカウンタ５２の桁あふれを抑えながら符号中に１／２領域切捨て処理に伴う余分な符号ビットを生じさせないことも可能である。本実施の形態３では、一例として上記実施の形態２で説明したＥＮＣＯＤＥＲ処理を説明するフロー（図１４）、ＤＥＣＯＤＥＲ処理を説明するフロー（図１８）を図２０、図２１に置き換えたものとする。

【０１３３】図２０は、符号化処理の全体の流れを示すＥＮＣＯＤＥＲ処理を説明するフローチャートである。

図１４におけるストライプの終了判定（Ｓ１０４）と１／２領域切捨て処理を行うＨＡＬＶＩＮＧＥ処理の呼出し（Ｓ１０７）の間にストライプの符号化処理中に正規化処理が行われた（符号ビットが発生したか）どうかの判定（Ｓ１０８）を追加し、行われた場合のみ１／２領域切捨て処理を行う。

【０１３４】図２１は、復号処理の全体の流れを示すＤＥＣＯＤＥＲ処理を説明するフローチャートである。図１８におけるストライプの終了判定（Ｓ２１４）と１／２領域切捨て処理を行うＨＡＬＶＩＮＧＤ処理の呼出し

（Ｓ２１６）の間にストライプの復号処理中に正規化処理が行われた（符号ビットが読み出された）かどうかの判定（Ｓ２１７）を追加し、行われた場合のみ１／２領域切捨て処理を行う。

【０１３５】ここで、符号レジスタＣ３０Ａの構成仕様を上記実施の形態１と同様にすれば、符号器／復号器のバイト入出力が同期可能なため、本実施の形態３において正規化処理の有無による１／２領域切捨て処理実行をライン処理中の符号バイト入出力の有無（実行有無フラグを導入または符号ビット数の増加から検出するなど）で行うことも可能である。この場合、バイト入出力までの数ビット未満の正規化処理が発生しても１／２領域切捨て処理の実行が行われないことになり、それに伴う余分な符号ビットを削減できる。

【０１３６】実施の形態４．上記実施の形態２では、画像をストライプに分割して符号化／復号する際に、１回のみ符号器／復号器の初期化処理および後処理を行うものとして示した。本実施の形態４では、１／２領域切捨て処理をラインの処理ごとに実行し、ストライプの処理ごとに符号器／復号器の初期化処理および後処理を行うものとする。ここで、符号器の後処理（ＦＬＵＳＨ処理を説明するフロー）には、未確定の符号を確定して掃き出す処理が行われるため、ストライプは複数ラインから構成されるものとして、ライン終端ごとに１／２領域切捨て処理を行うものとする。本実施の形態４では、一例として上記実施の形態２で説明したＥＮＣＯＤＥＲ処理を説明するフロー（図１４）、ＤＥＣＯＤＥＲ処理を説明するフロー（図１８）を図２２、図２３に置き換えたものとする。

【０１３７】図２２は、符号化処理の全体の流れを示すＥＮＣＯＤＥＲ処理を説明するフローチャートである。

図１４に対してラインの終了判定（Ｓ１０９）を追加したことにより、ストライプの終了判定（Ｓ１０４）、画像の終了判定（Ｓ１０５）を再構成することによって、ラインごとに１／２領域切捨て処理（Ｓ１０７）を実行し、ストライプごとに初期化処理（Ｓ１０１）、後処理（Ｓ１０６）を実行している。

【０１３８】図２３は、復号処理の全体の流れを示すＤＥＣＯＤＥＲ処理を説明するフローチャートである。図１８に対してラインの終了判定（Ｓ２１８）を追加したことにより、ストライプの終了判定（Ｓ２１４）、画像の終了判定（Ｓ２１５）を再構成することによって、ラインごとに１／２領域切捨て処理（Ｓ２１６）を実行し、ストライプごとに初期化処理（Ｓ２１１）を実行している。

【０１３９】このように、ラインごとに１／２領域切捨て処理を行うことによって、上記実施の形態２と同様に符号の確定遅延は最大８ラインとなり、かつ初期化処理、後処理を行うことによって符号がストライプごとに完結することになる。

【０１４０】実施の形態５．上記実施の形態３では、画像をストライプに分割して符号化／復号する際に、１回のみ符号器／復号器の初期化処理および後処理を行うものとして示した。本実施の形態５では、１／２領域切捨て処理をラインの処理ごとに実行し、ストライプの処理ごとに符号器／復号器の初期化処理および後処理を行うものとする。本実施の形態５では、一例として上記実施の形態２で説明したＥＮＣＯＤＥＲ処理を説明するフロー（図２０）、ＤＥＣＯＤＥＲ処理を説明するフロー（図２１）を、図２４、図２５に置き換えたものとする。

【０１４１】図２４は、符号化処理の全体の流れを示すＥＮＣＯＤＥＲ処理を説明するフローチャートである。

図２０に対してラインの終了判定（Ｓ１０９）を追加したことにより、ストライプの終了判定（Ｓ１０４）、画像の終了判定（Ｓ１０５）を再構成することによって、ラインごとに１／２領域切捨て処理（Ｓ１０７）を実行し、ストライプごとに初期化処理（Ｓ１０１）、後処理（Ｓ１０６）を実行している。

【０１４２】図２５は、復号処理の全体の流れを示すＤＥＣＯＤＥＲ処理を説明するフローチャートである。図２１に対してラインの終了判定（Ｓ２１８）を追加したことにより、ストライプの終了判定（Ｓ２１４）、画像の終了判定（Ｓ２１５）を再構成することによって、ラインごとに１／２領域切捨て処理（Ｓ２１６）を実行し、ストライプごとに初期化処理（Ｓ２１１）を実行している。

【０１４３】このように、ラインごとに１／２領域切捨て処理を行うことによって、上記実施の形態３と同様に符号の確定遅延は上記実施の形態４より大きくなるが、ＳＣカウンタ５２の桁あふれを抑えながら符号中に余分な符号ビットを生じさせないことになる。また、上記実施の形態４と同様に初期化処理、後処理を行うことによって符号がストライプごとに完結することになる。

【０１４４】上記実施の形態２から上記実施の形態５は、符号器と復号器における１／２領域切捨て処理の同期をストライプ、ラインといった単位で行っているが、例えばブロック、サブブロックといった他の何らかの同期のとれる単位であっても構わない。また、データは必ずしも画像でなくとも構わない。

【０１４５】実施の形態６．本実施の形態６では、例えばバイト単位で符号を出力する場合に、符号中に同じバイト値が連続するときそのバイト値の連続を、バイト値およびその長さを意味する他のバイト群で置き換えることによって総符号長とその出力時間を短縮する方法を示す。以下、ここで置き換えるバイト群をランレングスマーカを呼び、所定の指示方法によって任意の長さに拡張して数値を表現することができるものとする。

【０１４６】図２６に示すように、例えば４つのバイト０ｘＦＦの連続を、（１）ではエスケープコードＥＳＣ

とそれに続く０ｘＦＦが４つ続くことを示すバイトＭＫで置き換えることにより、ＥＳＣコードを確保するために挿入されたＳＴＵＦＦバイトと併せて８バイトを２バイトとする例である。（２）ではバイトＭＫは０ｘＦＦの置き換えであることのみ示し、その長さをバイトＲＬの３バイトに置き換える例である。また、０ｘ００が続く場合も同様に０ｘ００が続くことを示すバイトＭＫと別途必要ならばバイトＲＬで置き換える例である。ここで、バイトＭＫはその置き換える対象のバイト値およびその長さを識別できる値が割り振られているものとする。

【０１４７】また、図２７に示すように、７個のバイト０ｘＦＦを置き換えたランレングスマーカは、一例として、４個と３個の置換えを示す２つのランレングスマーカ、または２個と３個と２個の置換えを示す２つのランレングスマーカとも等価であり、また他のラン長の合計が等しくなる組合せであっても等価であるといえる。

【０１４８】ランレングスマーカは、例えば図２８に示すように、ＥＳＣコード（０ｘＦＦ）とそれに続くバイトＭＫで示すとすれば、バイトＭＫは（８－Ｎ）ビットの置換対象バイトの識別部とＮビットのランレングス部から構成される。この場合、置換対象バイトは２の（８－Ｎ）乗種類を指定でき、また２のＮ乗までの置換対象バイトのラン長を示すことができる。

【０１４９】ただし、国際標準勧告Ｔ．８２では他のマーカセグメントが定義されているため、識別部で指定可能なすべての値を使用できるわけではない。また、他の国際標準勧告でも別途マーカセグメントが使用されている可能性があるため、ここでランレングスマーカを使用する際にはそれらのすべてのマーカセグメントと重複しないようにしなければならない。

【０１５０】また、ランレングスマーカは、例えば図２９に示すように、ＥＳＣコード（０ｘＦＦ）とそれに続くバイトＭＫ、および所定のＳバイトのランレングス部ＲＬで示すとすれば、バイトＭＫは８ビットすべてが置換対象バイトの識別部となる。この場合、置換対象バイトは２５５種類（ＭＫ＝ＳＴＵＦＦを除く）を指定でき、また２の（８×Ｓ）乗までの置換対象バイトのラン長を示すことができる。

【０１５１】さらに、ランレングスマーカは、例えば図３０に示すように、ＥＳＣコード（０ｘＦＦ）とそれに続くバイトＭＫ、およびＳバイトのランレングス部ＲＬで示すとすれば、バイトＭＫは（８－Ｎ）ビットの置換対象バイトの識別部とＮビットのランレングス指示部から構成される。この場合、置換対象バイトは２の（８－Ｎ）乗種類を指定でき、また２のＮ乗バイトまでのランレングス部のバイト長を示すことができる。そして、２の（８×Ｓ）乗までの置換対象バイトのラン長を示すことができる。

【０１５２】図２８から図３０にランレングスマーカの一例を示したが、バイトＭＫで置換対象バイトを指示できないとき、ＥＳＣおよびバイトＭＫでランレングスマーカの開始を判定させ、固定長またはバイトＭＫで指定した長さのランレングス部に、さらに置換対象バイトをランレングス部の前または後に置くことで直接指定する方法もある。この方法によれば、他のマーカセグメントに対して、ランレングスマーカであることだけをバイトＭＫで認識させれば、置換対象バイトはその値によらずすべてが使用できるようになる。

【０１５３】また、複数バイトの繰返しに対しても、例えばバイトＭＫにその繰り返される置換対象パターンのバイト長を示し、パターンのラン長（繰返し数）とともにパターンを直接指定する方法もある。

【０１５４】一例を図３１（ａ）に示すと、ＥＳＣコード（０ｘＦＦ）と識別部を含むマーカＭＫ、複数バイトから構成される置換対象パターン部ＰＴ、ランレングス部ＲＬから構成されるものとする。ここで、マーカＭＫは、置換対象パターン部ＰＴのフィールド長（値Ｐ）、ランレングス部ＲＬのフィールド長（値Ｒ）を指定するとすればＰバイトの置換対象パターンＰＴがＲバイトで表現されるＲＬ回繰り返されることになる。

【０１５５】その具体例として、３回繰り返される２バイトのパターン０ｘ０１，０ｘ２３をランレングスマーカに置き換えると、置換対象パターン部のフィールド長Ｐが２、ランレングス部のフィールド長Ｒが１であるマーカＭＫを使用して、図３１（ｂ）のように示される。

【０１５６】マーカＭＫにおいて、ランレングスマーカ識別部を除く上記２つのフィールド長部は、その識別部を含むバイトと独立であっても構わない。また、置換対象パターン部ＰＴのフィールド長部とランレングス部ＲＬのフィールド長部、さらに置換対象パターン部ＰＴとランレングス部ＲＬは、符号器と復号器の間で認識が一致すれば上記一例に限らずそれぞれ自由に配置することができる。

【０１５７】ランレングスマーカに指示される置換対象バイトのラン長は０であることはないため、０が最大値を表すか、または実際の（ラン長－１）を常にランレングス部に埋め込むかを取り決めておく必要がある。

【０１５８】ここで、符号器における０ｘＦＦまたは０ｘ００の連続したＳＣ回のバイト出力（ＢＹＴＥＯＵＴ処理を説明するフローチャートのＳ１５４、Ｓ１６２、ＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフローチャートのＳ２０４、Ｓ２０８、ＣＯＤＥＬＯＷＥＲ処理を説明するフローチャートのＳ５２４、ＣＯＤＥＵＰＰＥＲ処理を説明するフローチャートのＳ５３６）については、例えば図３２に示すように、ＲＵＮＬＥＮＭＡＲＫ処理を説明するフローチャートでランレングスマーカへの置換え処理を容易に実行することができる。ここで、符号器と復号器では、上記図２８から図３０のどの形式のランレングスマーカを使用するかということを予め取り決

めているものとする。その使用するランレングスマーカのランレングス部に示すことのできる最大値を定数ＭＡＲＫＬＥＮ８０とする。例えば、図２８において、Ｎ＝４であれば、定数ＭＡＲＫＬＥＮ８０は１６となる。

【０１５９】図３２において、ＲＬ８１はＳＣ個５２のバイトを置き換えるためのカウンタ変数である。Ｓ６０１では、変数ＳＣ値５２を変数ＲＬ８１に設定する。Ｓ６０２で変数ＲＬ値８１が定数ＭＡＲＫＬＥＮ８０より大きければ、Ｓ６０３で長さＭＡＲＫＬＥＮ８０に対する置換対象バイトを示すランレングス（ＲＬ）マーカを書き出し、Ｓ６０４で変数ＲＬ８１から定数ＭＡＲＫＬＥＮ８０を減じる。Ｓ６０２で変数ＲＬ値８１が定数ＭＡＲＫＬＥＮ８０より大きくなくなれば、最後にＳ６０５で残りの長さが変数ＲＬ値８１の置換対象バイトを示すランレングス（ＲＬ）マーカを書き出し、Ｓ６０６で変数ＲＬ値８１を０とする。

【０１６０】この処理を説明するフローチャートによれば、ランレングスマーカが分割されて連続出力されるとき最大長のものから出力され、最後に最大値未満のものが出力されることになるが、上記図２７にも示したとおりこの出力順によらなくてもよい。また、置換対象バイトは必ずしも０ｘ００または０ｘＦＦである必要はなく、符号出力を監視することによって任意の連続する符号バイトを置き換えてもよい。

【０１６１】ランレングスマーカのランレングス部のビット数が、ＳＣカウンタ５２のビット数以上であれば、ラン長の指示についてただ１つのランレングスマーカで指定することが可能である。

【０１６２】置換対象バイトは、必ずしもバイトであることはなく、例えばワード、ダブルワードといったマルチバイト単位でもよく、置き換えるランレングスマーカを適正に選択することによって、１バイトであってもよいし、複数から構成されるパターンを対象とすることもできる。

【０１６３】実施の形態７．上記実施の形態６によれば、ラン長が変数ＳＣ値５２の置換対象バイトはすべてランレングスマーカで置き換えられることになる。本実施の形態７では、図３３において、一部のランレングス分だけランレングスマーカで置き換えずに置換対象バイトをその長さだけそのまま出力できるものとする。ここで、ランレングスマーカの構成バイト数を定数ＭＡＲＫＬＥＮＭＩＮ８２とする。例えば、図２８であれば定数ＭＡＲＫＬＥＮＭＩＮは２となる。

【０１６４】図３４は、ランレングスマーカの置換え処理を行うＲＵＮＬＥＮＭＡＲＫ処理を説明するフローチャートである。図３４において、Ｓ６０１からＳ６０４およびＳ６０６は上記図３２と同様の処理が行われる。Ｓ６０７において変数ＲＬ値８１が定数ＭＡＲＫＬＥＮＭＩＮ８２以上であれば、Ｓ６０５で長さが変数ＲＬ値８１の置換対象バイトを示すランレングス（ＲＬ）マーカを書き出し、定数ＭＡＲＫＬＥＮＭＩＮ８２未満であればＳ６０８で置換対象バイトを変数ＲＬ値８１で示される個数だけ書き出す。

【０１６５】このように、置換対象バイトのラン長がランレングスマーカの構成バイト数より少なければランレングスマーカへ置き換えずに置換対象バイトをそのランの長さ分だけそのまま出力することによって符号長を短くできる。どちらで出力しても長さが等しい場合には、置き換えても置き換えなくてもよい。

【０１６６】ただし、置換対象バイトがバイト０ｘＦＦである場合には、置き換えないときＳＴＵＦＦバイトの挿入が行われて符号長が２倍となるため、より短くなるように長さの判定を考慮すべきである。

【０１６７】実施の形態８．上記実施の形態６あるいは上記実施の形態７において、ランレングスマーカを使用する際には、図３５に示すように、出力される符号中に使用を通知する情報を含ませることができる。

【０１６８】常に使用する場合はランレングスマーカの使用通知は行う必要はないが、例えば上記図２８から図３０などのランレングスマーカの形式、置換対象バイトの種別、ランレングス部の長さの制限などの一部またはすべてについて、符号器から復号器へ付加情報を予め通知することによって動作モードの統一設定を図ることができる。

【０１６９】また、符号器は、符号化に先立って復号器に実装される処理能力を確認し（ネゴシエーション）、その結果から適用したモード情報を通知してもよい。復号器の能力としては、例えばランレングスマーカを逆変換するためのランレングス部のカウンタ桁数などの制限値がある。

【０１７０】さらに、この場合、復号器は、ランレングスマーカ使用通知を含まない符号データに対しては、従来の符号器の生成した符号としてみなすこともできる。また、符号器は、復号器が従来の復号器であれば、ランレングスマーカ使用通知を符号に挿入せず、またランレングスマーカを使用しなければよい。

【０１７１】実施の形態９．上記実施の形態６あるいは上記実施の形態７に示されたランレングスマーカを従来の符号器および従来の復号器で使用するために、従来の符号器と回線（通信路）の間、または回線（通信路）と従来復号器の間にアダプタを挿入することによって相互通信を可能とすることができる。

【０１７２】まず、図３６に示すように、符号器側のアダプタ（ランレングスマーカ変換器）は、データから符号を区別し、符号バイトを監視しながら同一バイト値が連続する場合、そのラン長を改めてＳＣとすれば、図３２または図３４の処理を説明するフローチャートによってランレングスマーカへ置き換える。また、復号器側のアダプタ（ランレングスマーカ逆変換器）は、同様にデータから符号を区別し、符号バイトを監視しながらラン

レングスマーカを検出した場合指示された置換対象バイトを検出されたラン長だけ連続して出力する。

【０１７３】また、図３７に示すように、復号器側にランレングスマーカ逆変換器がないとき、符号器はランレングスマーカ変換器を実質的に使用せずにそのまま符号を通過させ、図３８に示すように、符号器側にランレングスマーカ変換器がないとき、復号器はランレングスマーカ逆変換器を実質的に使用せずにそのまま符号を通過させることによってランレングスマーカを使用できない従来装置との互換性を保持することができる。

【０１７４】ここで、従来の符号器／復号器に接続されるアダプタ（ランレングス変換器／逆変換器）は、外見が符号器／復号器に対して独立構成であってもよいし、一体構成であっても構わない。

【０１７５】以上のように、符号化および復号処理は、計算機または装置の内部でデータを蓄積したり、公衆回線または専用回線による無線・有線による通信または記憶媒体によって、外部とのデータの交換を行うファクシミリ装置、スキャナ装置、プリンタ装置、コンピュータ、データベース装置、画像表示装置、画像蓄積装置、データ蓄積装置、画像伝送装置、データ伝送装置等のハードウェアに適用することができる。また、汎用の計算機にソフトウェアと一部のハードウェアで上記各装置相当の機能を実現することもでき、装置の外見が一体であっても複数に独立していても構わない。さらに、専用のハードウェア内部についても、これらの処理機能を搭載したＬＳＩ（半導体チップ）やミドルウェアによる実装形式には制限されるものではない。通信についても、電気的または光学的無線や有線、また公衆回線や専用回線、あるいはＬＡＮ、ＷＡＮ、インターネット、イントラネットなどの通信形式に制限されるものではない。記憶媒体への記録形式も磁気的、電気的、光学的あるいはデジタル式やアナログ式といった記録形式に制限されるものでもなく、また、記録媒体は装置に固定されるか分離されてるかに制限されるものでもない。さらに、視認可否に関わらずインク等での記録形式でも構わない。

【０１７６】なお、この発明において、ブロック図、フロー図は画像符号化に適用するものとして説明を行ったが、一般的なデータの符号化にも用いることができることはいうまでもない。

【０１７７】

【発明の効果】以上のように、この発明によれば、符号が確定するまで待機する符号長を記憶するカウンタの桁あふれをなくすことができ、適当な間隔で符号を強制的に確定することにより桁上がりを制御して符号の確定遅延時間を有限化して見積もることができる。また、符号値が確定したときに所定のパターンへの置き換えによって総符号長と符号の掃出し時間を短縮することができる。

【０１７８】また、桁上がり境界を厳密に計算せずに桁上がり制御を行うことができると共に、復号器で符号器の符号レジスタを再現せずに桁上がり制御を行うことができるという効果が得られる。

【図面の簡単な説明】

【図１】　この発明の概念に係る上方／下方１／２領域切捨て方式の説明図である。

【図２】　この発明の実施の形態１に係る符号化部の構成を示すブロック図である。

【図３】　この発明の実施の形態１に係る復号化部の構成を示すブロック図である。

【図４】　この発明の実施の形態１に係るＲＯＵＮＤＤＯＷＮＥ処理を説明するフローチャートである。

【図５】　この発明の実施の形態１に係るＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。

【図６】　この発明の実施の形態１に係るＲＯＵＮＤＤＯＷＮＤ処理を説明するフローチャートである。

【図７】　この発明の実施の形態１に係るＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。

【図８】　この発明の実施の形態１に係るＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。

【図９】　この発明の実施の形態１に係るＢＹＴＥＯＵＴ７処理を説明するフローチャートである。

【図１０】　この発明の実施の形態１に係るＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。

【図１１】　この発明の実施の形態１に係るＢＹＴＥＩＮ７処理を説明するフローチャートである。

【図１２】　この発明の実施の形態２に係る符号化部の構成を示すブロック図である。

【図１３】　この発明の実施の形態２に係る復号化部の構成を示すブロック図である。

【図１４】　この発明の実施の形態２に係るＥＮＣＯＤＥＲ処理を説明するフローチャートである。

【図１５】　この発明の実施の形態２に係るＨＡＬＶＩＮＧＥ処理を説明するフローチャートである。

【図１６】　この発明の実施の形態２に係るＣＯＤＥＬＯＷＥＲ処理を説明するフローチャートである。

【図１７】　この発明の実施の形態２に係るＣＯＤＥＵＰＰＥＲ処理を説明するフローチャートである。

【図１８】　この発明の実施の形態２に係るＤＥＣＯＤＥＲ処理を説明するフローチャートである。

【図１９】　この発明の実施の形態２に係るＨＡＬＶＩＮＧＤ処理を説明するフローチャートである。

【図２０】　この発明の実施の形態３に係るＥＮＣＯＤＥＲ処理を説明するフローチャートである。

【図２１】　この発明の実施の形態３に係るＤＥＣＯＤＥＲ処理を説明するフローチャートである。

【図２２】　この発明の実施の形態４に係るＥＮＣＯＤＥＲ処理を説明するフローチャートである。

【図２３】　この発明の実施の形態４に係るＤＥＣＯＤＥＲ処理を説明するフローチャートである。

【図２４】　この発明の実施の形態５に係るＥＮＣＯＤＥＲ処理を説明するフローチャートである。
【図２５】　この発明の実施の形態５に係るＤＥＣＯＤＥＲ処理を説明するフローチャートである。
【図２６】　この発明の実施の形態６に係るランレングスマーカの説明図である。
【図２７】　この発明の実施の形態６に係るランレングスマーカ分割出力の説明図である。
【図２８】　この発明の実施の形態６に係るランレングスマーカの一例を示す構成図である。
【図２９】　この発明の実施の形態６に係るランレングスマーカの一例を示す構成図である。
【図３０】　この発明の実施の形態６に係るランレングスマーカの一例を示す構成図である。
【図３１】　この発明の実施の形態６に係るバイトＭＫに置換対象パターンのバイト長を示し、パターンのラン長とともにパターンを直接指定する方法の説明図である。
【図３２】　この発明の実施の形態６に係るＲＵＮＬＥＮＭＡＲＫ処理を説明するフローチャートである。
【図３３】　この発明の実施の形態７に係るランレングスマーカ符号長の比較に関する説明図である。
【図３４】　この発明の実施の形態７に係るＲＵＮＬＥＮＭＡＲＫ処理を説明するフローチャートである。
【図３５】　この発明の実施の形態８に係るランレングスマーカ適用を通知する情報の説明図である。
【図３６】　この発明の実施の形態９に係るランレングスマーカを使用して符号器と復号器間の相互通信を行う構成の一例を示すブロック図である。
【図３７】　この発明の実施の形態９に係るランレングスマーカを使用して符号器と復号器間の相互通信を行う構成の一例を示すブロック図である。
【図３８】　この発明の実施の形態９に係るランレングスマーカを使用して符号器と復号器間の相互通信を行う構成の一例を示すブロック図である。
【図３９】　従来の算術符号化の概念を示す説明図である。
【図４０】　小数部を常に領域分割の精度に保ち、上位桁を整数部に掃き出すための従来の正規化処理の説明図である。
【図４１】　第１の従来例に係る符号器の構成を示すブロック図である。
【図４２】　第１の従来例に係る復号器の構成を示すブロック図である。
【図４３】　標準モデルテンプレートの説明図である。
【図４４】　確率推定テーブルの説明図である。
【図４５】　符号化／復号／領域幅レジスタの構成図である。
【図４６】　第１の従来例に係るＥＮＣＯＤＥＲ処理を説明するフローチャートである。
【図４７】　第１の従来例に係るＥＮＣＯＤＥ処理を説明するフローチャートである。
【図４８】　第１の従来例に係るＣＯＤＥＬＰＳ処理を説明するフローチャートである。
【図４９】　第１の従来例に係るＣＯＤＥＭＰＳ処理を説明するフローチャートである。
【図５０】　第１の従来例に係るＲＥＮＯＲＭＥ処理を説明するフローチャートである。
【図５１】　第１の従来例に係るＢＹＴＥＯＵＴ処理を説明するフローチャートである。
【図５２】　第１の従来例に係るＩＮＩＴＥＮＣ処理を説明するフローチャートである。
【図５３】　第１の従来例に係るＦＬＵＳＨ処理を説明するフローチャートである。
【図５４】　第１の従来例に係るＣＬＥＡＲＢＩＴＳ処理を説明するフローチャートである。
【図５５】　第１の従来例に係るＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフローチャートである。
【図５６】　第１の従来例に係るＤＥＣＯＤＥＲ処理を説明するフローチャートである。
【図５７】　第１の従来例に係るＤＥＣＯＤＥ処理を説明するフローチャートである。
【図５８】　第１の従来例に係るＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。
【図５９】　第１の従来例に係るＭＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。
【図６０】　第１の従来例に係るＲＥＮＯＲＭＤ処理を説明するフローチャートである。
【図６１】　第１の従来例に係るＢＹＴＥＩＮ処理を説明するフローチャートである。
【図６２】　第１の従来例に係るＩＮＩＴＤＥＣ処理を説明するフローチャートである。
【図６３】　第２の従来例に係る桁上がり／非桁上がり領域切捨て処理の説明図である。
【図６４】　第２の従来例に係る符号器の構成を示すブロック図である。
【図６５】　第２の従来例に係る復号器の構成を示すブロック図である。
【図６６】　第２の従来例に係る符号化／領域幅レジスタの構成図である。
【図６７】　第２の従来例に係るＲＥＮＯＲＭＥ処理を説明するフローチャートである。
【図６８】　第２の従来例に係るＲＯＵＮＤＤＯＷＮＥ処理を説明するフローチャートである。
【図６９】　第２の従来例に係るＢＹＴＥＯＵＴ処理を説明するフローチャートである。
【図７０】　第２の従来例に係るＩＮＩＴＥＮＣ処理を説明するフローチャートである。
【図７１】　第２の従来例に係るＦＩＮＡＬＷＲＩＴＥＳ処理を説明するフローチャートである。

41

【図７２】　第２の従来例に係るＬＰＳ＿ＥＸＣＨＡＮＧＥ処理を説明するフローチャートである。

【図７３】　第２の従来例に係るＲＥＮＯＲＭＤ処理を説明するフローチャートである。

【図７４】　第２の従来例に係るＲＯＵＮＤＤＯＷＮＤ処理を説明するフローチャートである。

【図７５】　第２の従来例に係るＢＹＴＥＩＮ処理を説明するフローチャートである。

【図７６】　第２の従来例に係るＩＮＩＴＤＥＣ処理を説明するフローチャートである。

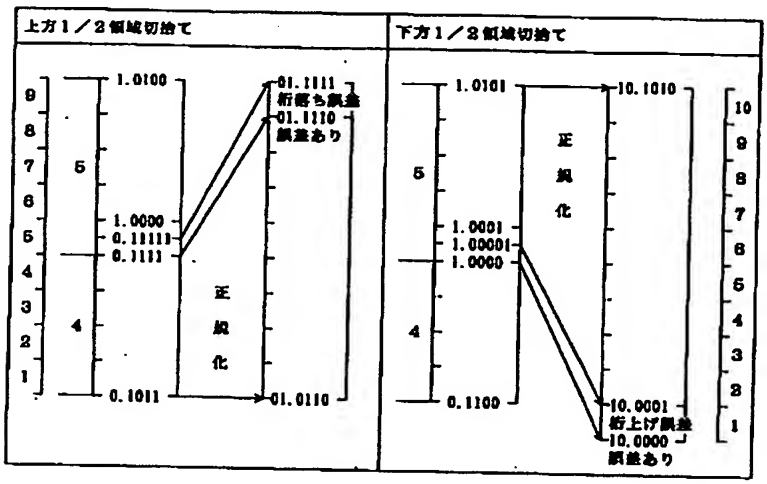
【符号の説明】

１Ａ　符号部、１Ｂ　復号部、２　コンテクストＣＸ、３　画素ＰＩＸ、４符号、５　データメモリ、６　画像、７　予測値テーブルＭＰＳ、８　ステートテーブルＳＴ、９　ＬＰＳ領域幅テーブルＬＳＺ、１０　ＭＰＳ状態遷移先テーブルＮＭＰＳ、１１　ＬＰＳ状態遷移先テーブルＮＬＰＳ、１２　予測値反転判定テーブルＳＷＴＣＨ、１３Ａ　算術符号器、１３Ｂ　算術復号器、１

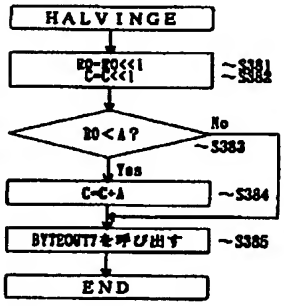
42

４　シンボル、１５Ａ　画素シンボル変換器、１５Ｂ　シンボル画素変換器、１６　予測値更新処理器、１７　ステート更新処理器、１８　ステートテーブルＳＴＡＴＥ、１９　非予測値テーブルＬＰＳ、３０Ａ　符号化レジスタ（下界値）Ｃ、３０Ｂ　復号レジスタＣ、３１　領域幅レジスタＡ、３２　Ｃｘレジスタ、３３　Ｃｓレジスタ、３４　Ｃｂレジスタ、３５　Ｃｃレジスタ、３６　ＣＬＯＷレジスタ、３７　Ｃｂレジスタ、３８　ＣＨＩＧＨレジスタ、３９　Ｃｘレジスタ、５０　ＣＴカウンタ、５１　符号バッファＢＵＦＦＥＲ、５２　ＳＣカウンタ、５３　出力判定バッファＴＥＭＰ、６０　上界値Ｕ、６１　桁上がり境界Ｔ、６２　桁上がり領域幅Ｒ１、６３　非桁上がり領域幅Ｒ０、６４　（復号器の）符号レジスタＤ、６５　１／２領域境界値Ｔ、６６　桁上がり境界値ＣＷ、８０　ランレングス部最大値ＭＡＲＫＬＥＮ、８１　ランレングスカウンタＲＬ、８２　ランレングスマーカ長ＭＡＲＫＬＥＮＭＩＮ。

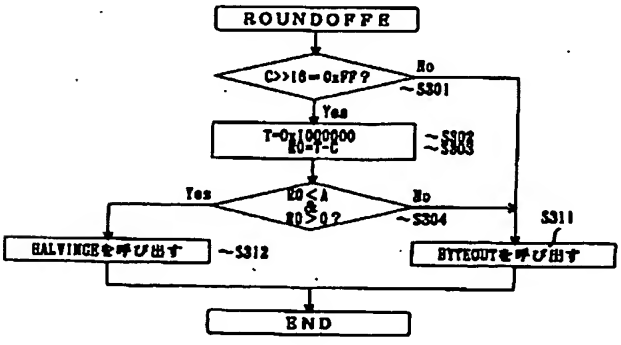
【図１】



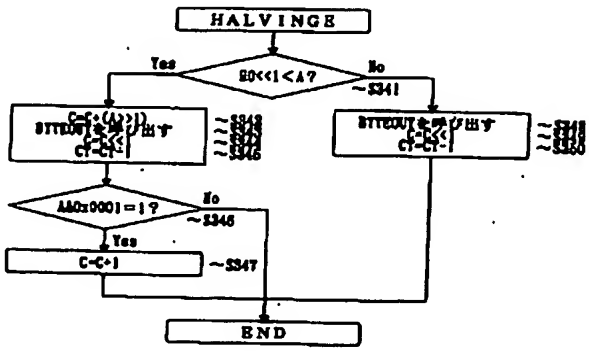
【図８】



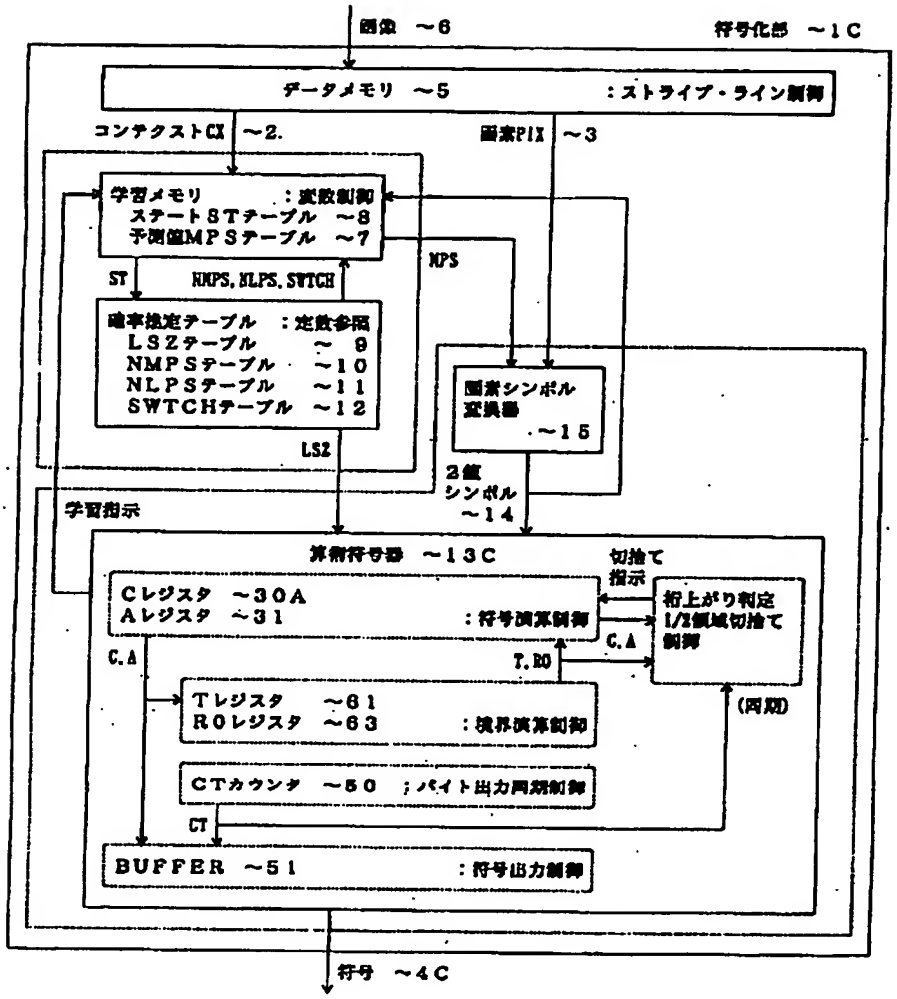
【図４】



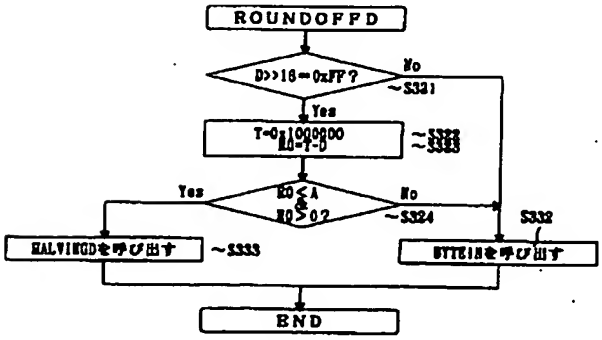
【図５】



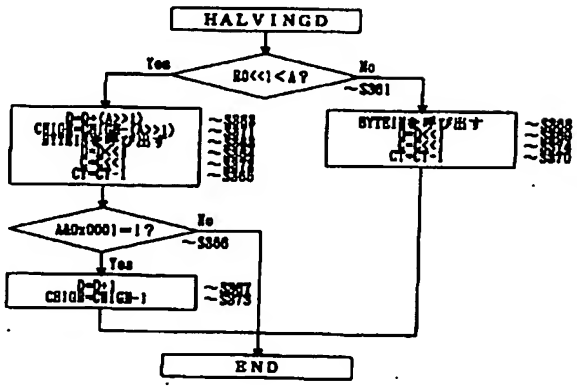
【図２】


画像　～６
符号化部　～１Ｃ
データメモリ　～５　：ストライプ・ライン制御
コンテクストCX　～２.
画素PIX　～３
学習メモリ　：変数制御
ステートSTテーブル　～８
予測値MPSテーブル　～７
MPS
ST
NMPS, NLPS, SWTCH
確率推定テーブル　：定数参照
ＬＳＺテーブル　～９
ＮＭＰＳテーブル　～１０
ＮＬＰＳテーブル　～１１
ＳＷＴＣＨテーブル　～１２
LSZ
画素シンボル変換器　～１５
２値シンボル　～１４
学習指示
算術符号器　～１３Ｃ
切捨て指示
Ｃレジスタ　～３０Ａ
Ａレジスタ　～３１　：符号演算制御
桁上がり判定 1/2領域切捨て制御
C.A
T.RO
(再現)
Ｔレジスタ　～６１
Ｒ０レジスタ　～６３　：境界演算制御
ＣＴカウンタ　～５０　；バイト出力同期制御
CT
BUFFER　～５１　：符号出力制御
符号　～４Ｃ

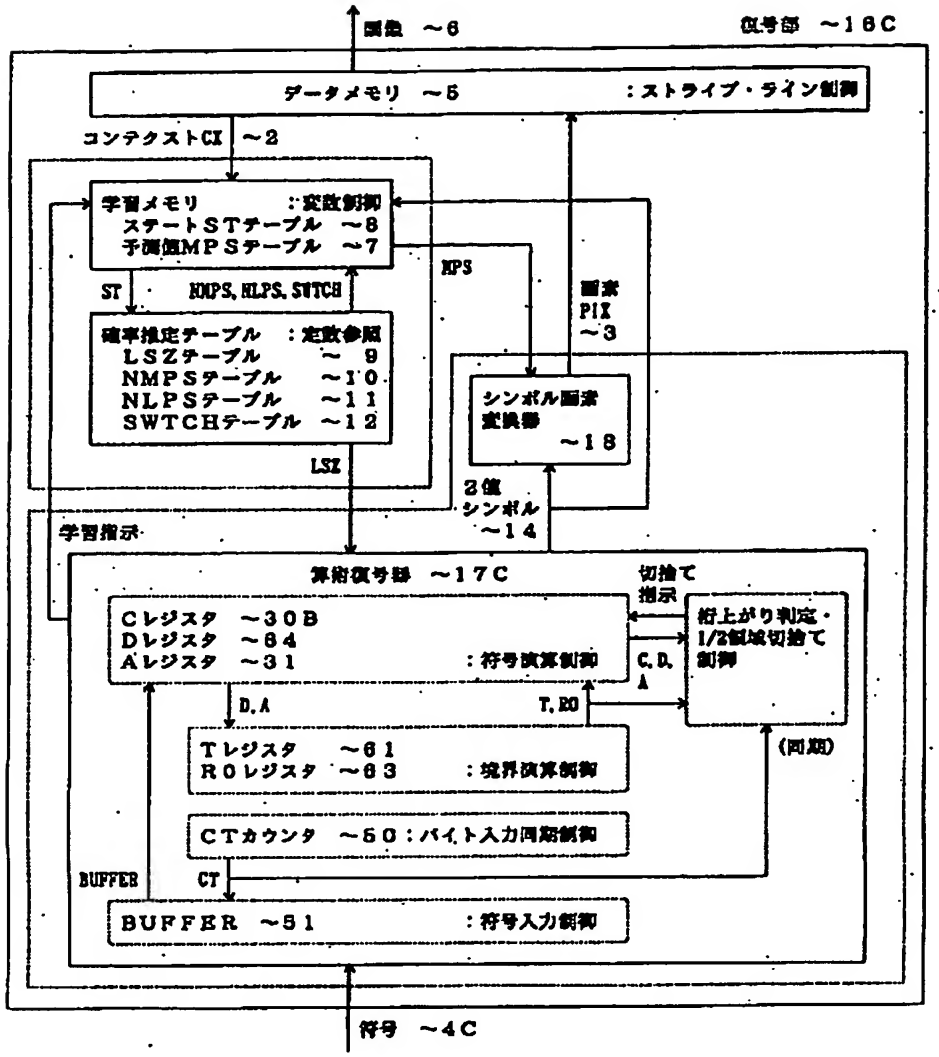
【図６】


ROUNDOFFD
D>>16=0xFF?
No
～S321
Yes
T=0x1000000
A=T-D
～S323
Yes
R0<A
&
R0>0?
No
～S324
S332
HALVINGDを呼び出す
～S333
BYTEINを呼び出す
END

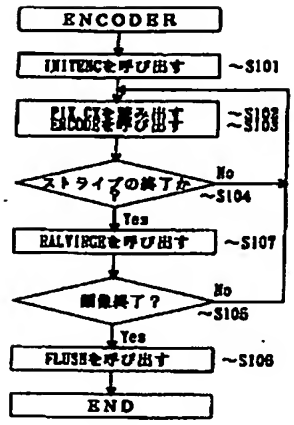
【図７】


HALVINGD
Yes
R0<<1<A?
No
～S361
CT=CT-1
BYTEINを呼び出す
CT=CT-1
A&0x0001=1?
No
～S366
Yes
CT=CT-1
～S375
END

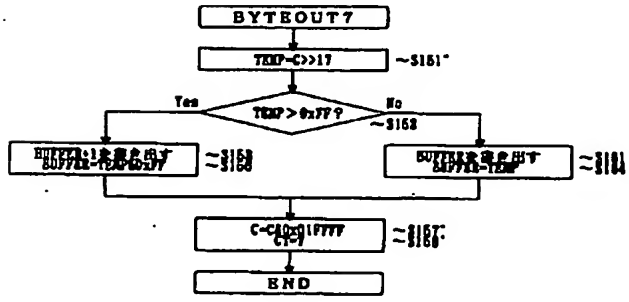
【図３】


画像　～６
復号部　～１６Ｃ
データメモリ　～５　：ストライプ・ライン制御
コンテクストＣＸ　～２
学習メモリ　：変数制御
ステートＳＴテーブル　～８
予測値ＭＰＳテーブル　～７
ＭＰＳ
ＳＴ
ＭＰＳ, ＬＰＳ, ＳＷＴＣＨ
確率推定テーブル　：定数参照
ＬＳＺテーブル　～　９
ＮＭＰＳテーブル　～１０
ＮＬＰＳテーブル　～１１
ＳＷＴＣＨテーブル　～１２
ＬＳＺ
画素
ＰＩＸ
～３
シンボル画素
変換器
～１８
２値
シンボル
～１４
学習指示
算術復号器　～１７Ｃ
切捨て
指示
Ｃレジスタ　～３０Ｂ
Ｄレジスタ　～８４
Ａレジスタ　～３１　：符号演算制御
桁上がり判定・
1/2領域切捨て
制御
Ｃ, Ｄ, Ａ
Ｄ, Ａ
Ｔ, ＲＯ
Ｔレジスタ　～６１
ＲＯレジスタ　～８３　：境界演算制御
（同期）
ＣＴカウンタ　～５０：バイト入力同期制御
ＢＵＦＦＥＲ
ＣＴ
ＢＵＦＦＥＲ　～５１　：符号入力制御
符号　～４Ｃ

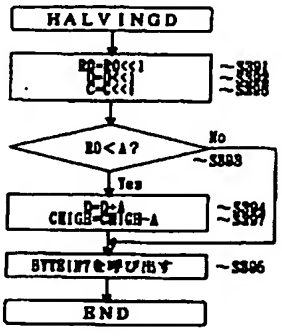
【図１４】


ＥＮＣＯＤＥＲ
ＩＮＩＴＥＮＣを呼び出す
～Ｓ１０１
ＰＩＬＣＴを書き出す
ＥＮＣＯＤＥを呼び出す
～Ｓ１０３
ストライプの終了か？
Ｎｏ
～Ｓ１０４
Ｙｅｓ
ＨＡＬＶＩＮＧＤを呼び出す
～Ｓ１０７
画像終了？
Ｎｏ
～Ｓ１０５
Ｙｅｓ
ＦＬＵＳＨを呼び出す
～Ｓ１０６
ＥＮＤ

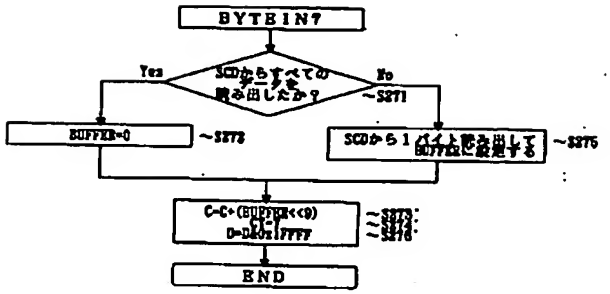
【図９】


ＢＹＴＥＯＵＴ７
ＴＥＭＰ=Ｃ>>１７
～Ｓ１５１
Ｙｅｓ
ＴＥＭＰ > ０ｘＦＦ？
Ｎｏ
～Ｓ１５２
～Ｓ１５３
～Ｓ１５４
～Ｓ１５５
～Ｓ１５６
ＥＮＤ

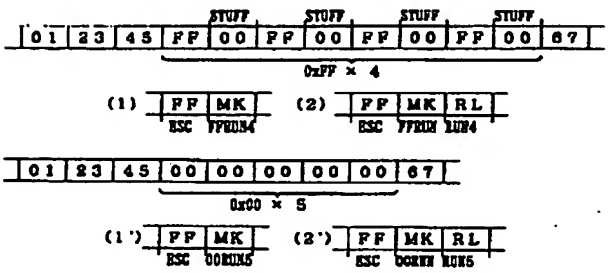
【図１０】


ＨＡＬＶＩＮＧＤ
～Ｓ３０１
～Ｓ３０２
～Ｓ３０３
Ｎｏ
Ｙｅｓ
～Ｓ３０４
～Ｓ３０７
～Ｓ３０６
ＥＮＤ

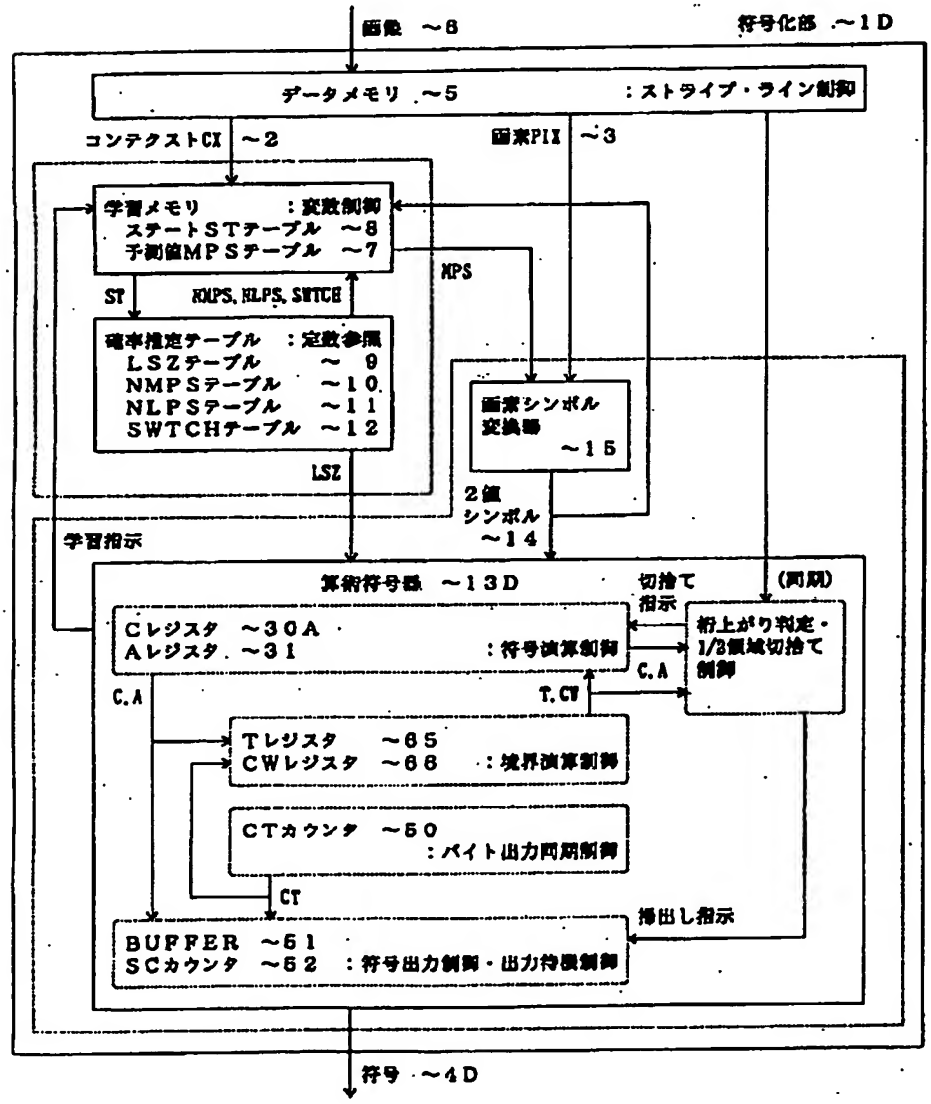
【図１１】


BYTEIN7
Yes
SCDからすべての
データを
読み出したか？
No
～S271
BUFFER=0
～S272
SCDから1バイト読み出して
BUFFERに設定する
～S275
C=C+(BUFFER<<8)
CT=7
D=D&0x7FFF
～S273
～S274
～S276
END

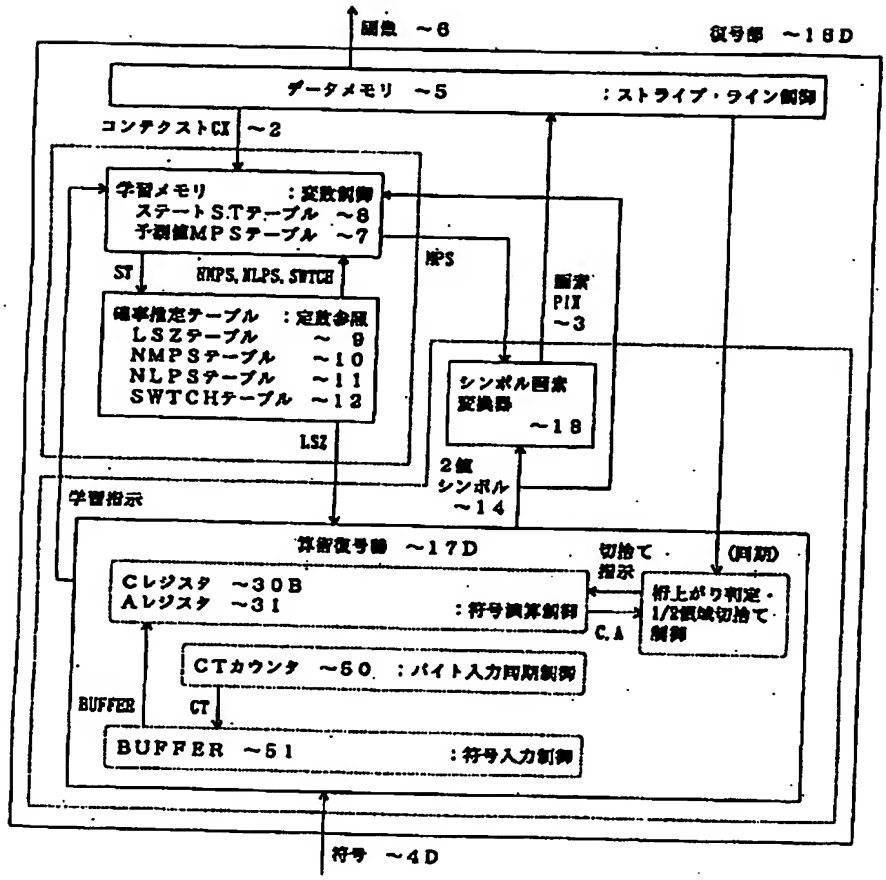
【図２６】


STUFF
STUFF
STUFF
STUFF
01 23 45 FF 00 FF 00 FF 00 FF 00 67
0xFF × 4
(1) FF MK
(2) FF MK RL
01 23 45 00 00 00 00 00 67
0x00 × 5
(1') FF MK
(2') FF MK RL

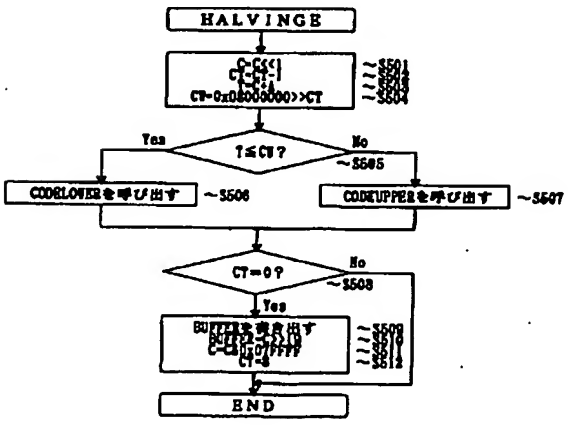
【図１２】


画像 ～8
符号化部 ～1D
データメモリ ～5
：ストライプ・ライン制御
コンテクストCX ～2
画素PIX ～3
学習メモリ
：変数制御
ステートSTテーブル ～8
予測値MPSテーブル ～7
MPS
ST
NMPS, NLPS, SWITCH
確率推定テーブル
：定数参照
LSZテーブル ～9
NMPSテーブル ～10
NLPSテーブル ～11
SWTCHテーブル ～12
LSZ
画素シンボル
変換器
～15
2値
シンボル
～14
学習指示
算術符号器 ～13D
切捨て
指示
(同期)
Cレジスタ ～30A
Aレジスタ ～31
：符号演算制御
桁上がり判定・
1/3領域切捨て
制御
C.A
T.CT
Tレジスタ ～65
CWレジスタ ～66
：境界演算制御
CTカウンタ ～50
：バイト出力同期制御
CT
掃出し指示
BUFFER ～51
SCカウンタ ～52
：符号出力制御・出力待機制御
符号 ～4D

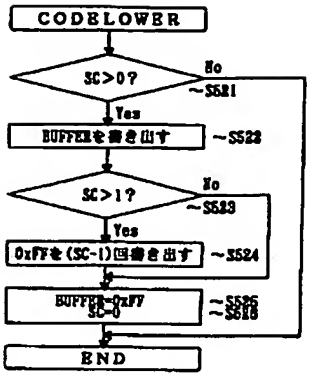
【図１３】



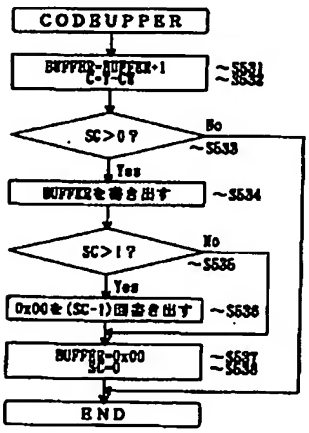
【図１５】



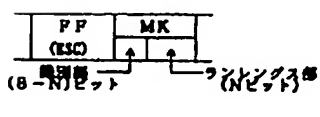
【図１６】



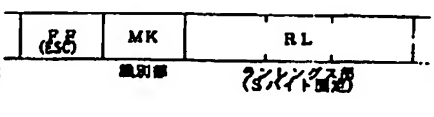
【図１７】



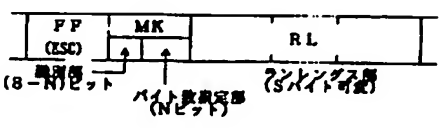
【図２８】



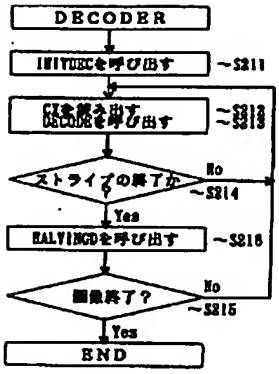
【図２９】



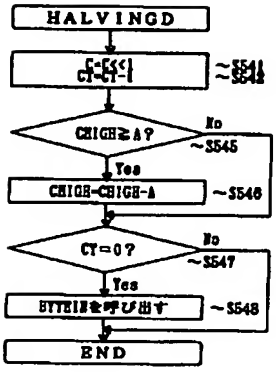
【図３０】



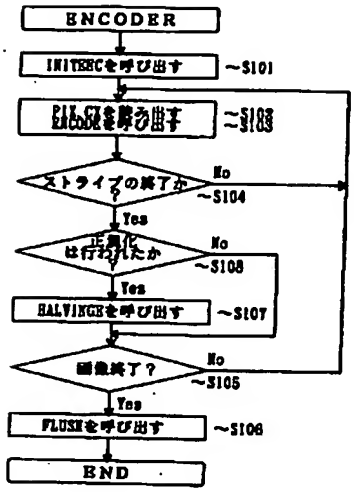
【図１８】



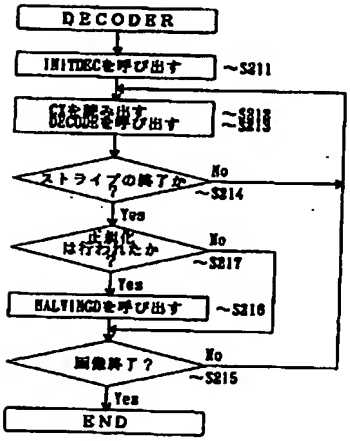
【図１９】



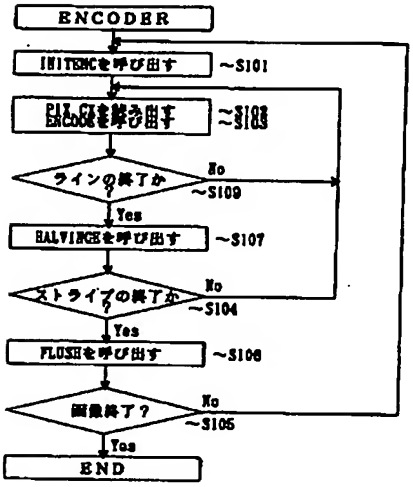
【図２０】



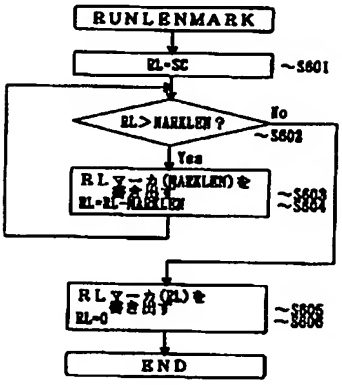
【図２１】



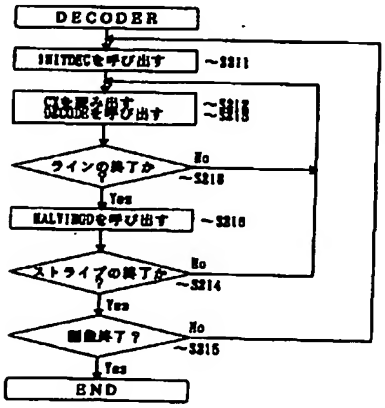
【図２２】



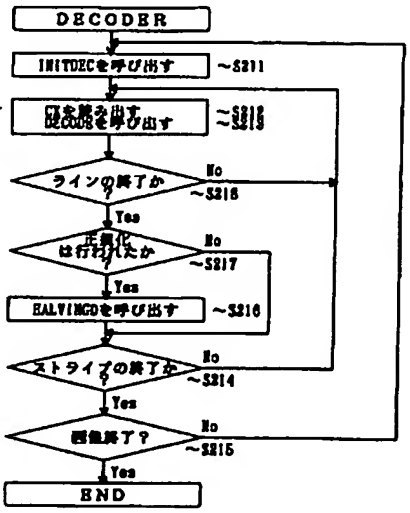
【図３２】



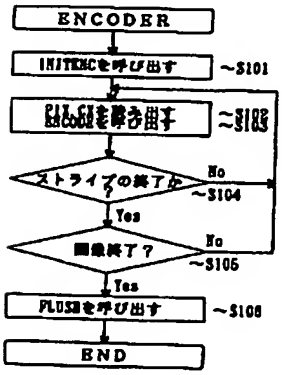
【図２３】



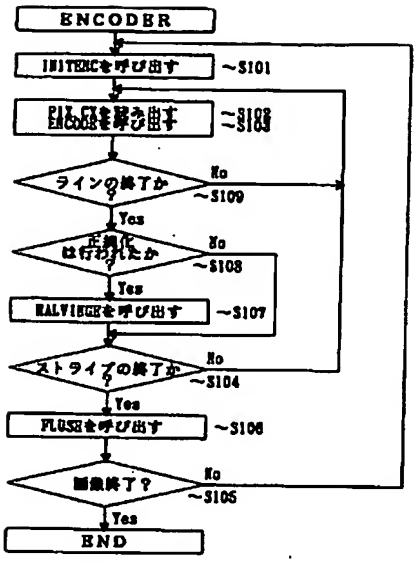
【図２５】



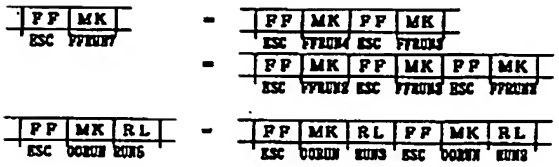
【図４６】



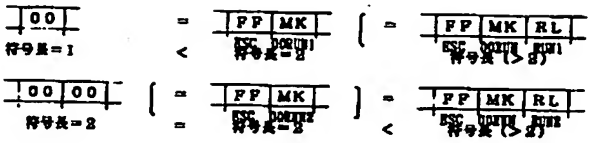
【図２４】



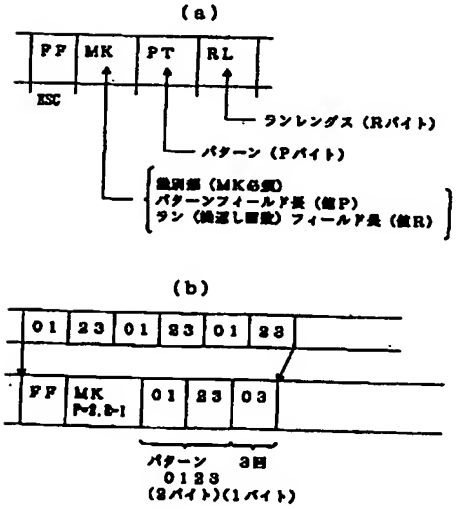
【図２７】



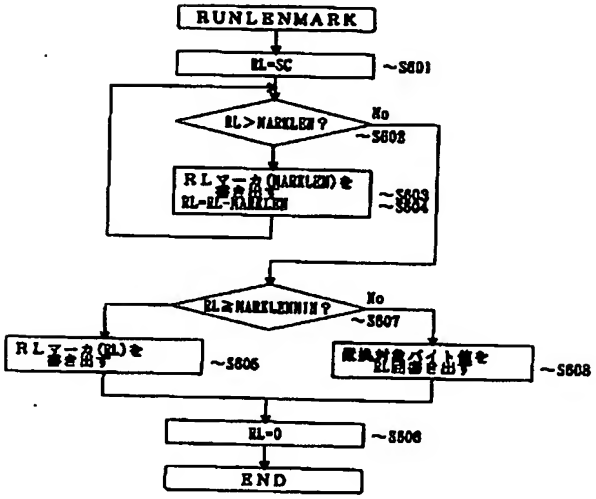
【図３３】



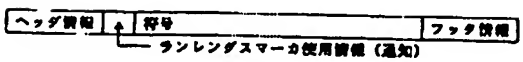
【図３１】



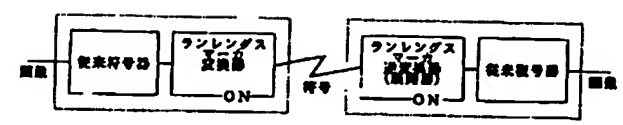
【図３４】



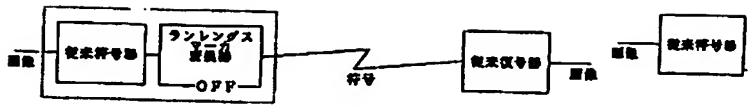
【図３５】



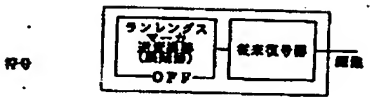
【図３６】



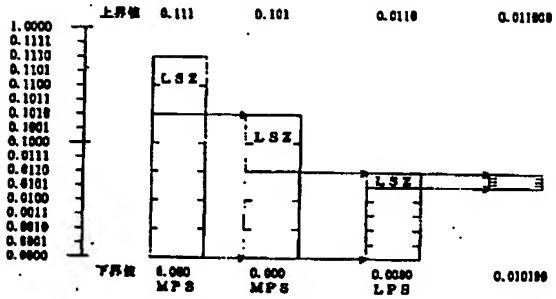
【図３７】



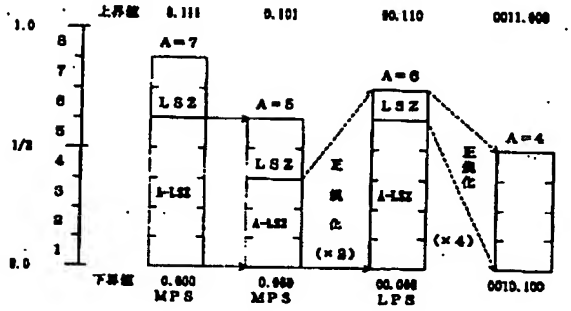
【図３８】



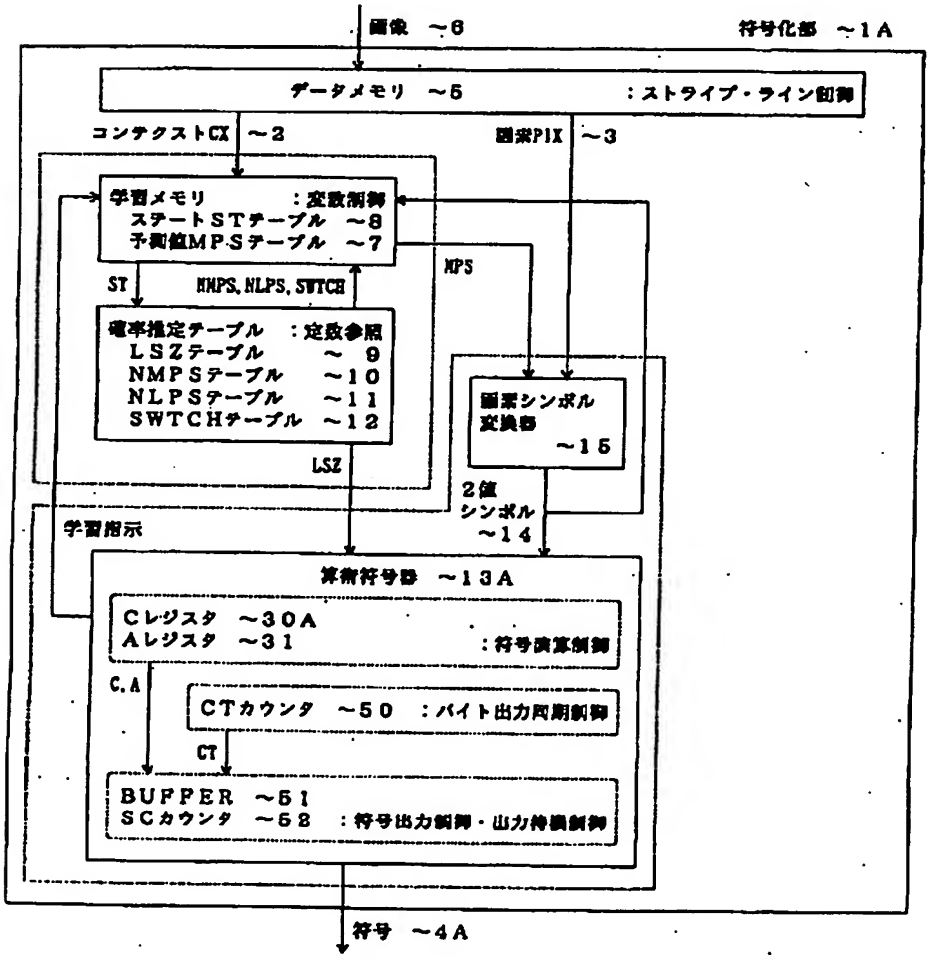
【図３９】



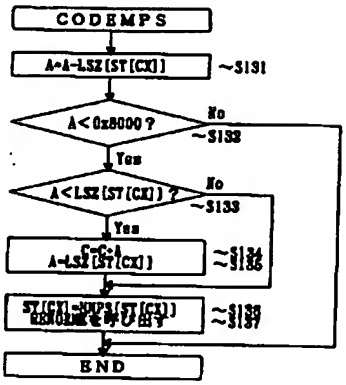
【図４０】



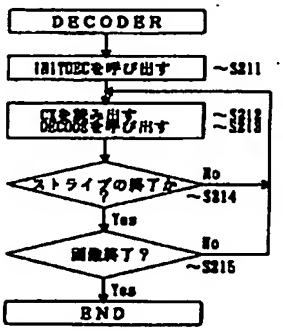
【図４１】



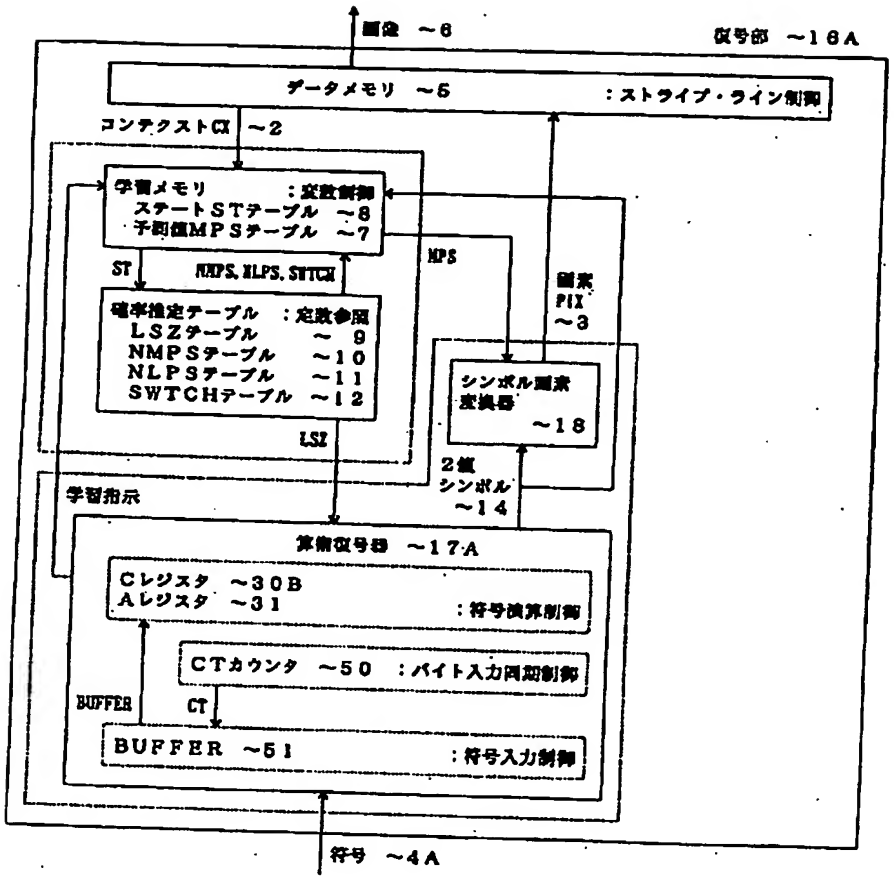
【図４９】



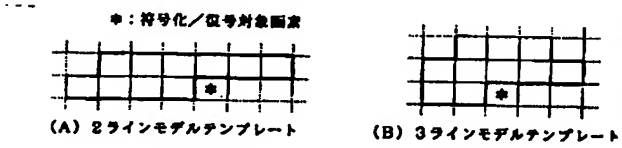
【図５６】



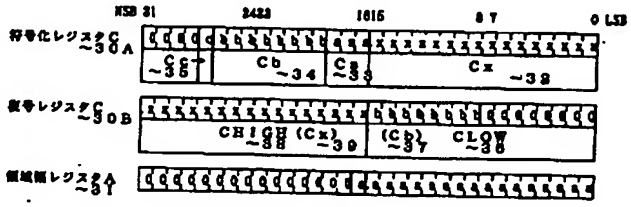
【図４２】



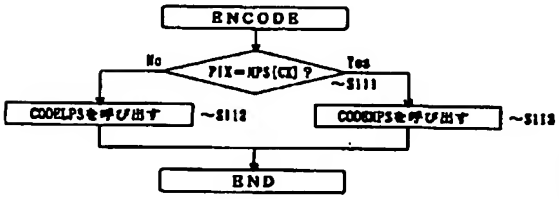
【図４３】



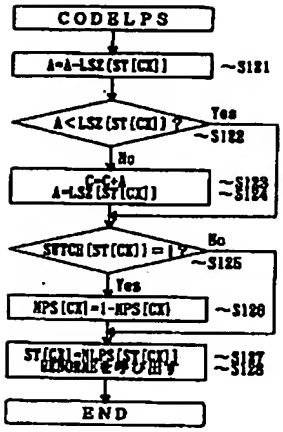
【図４５】



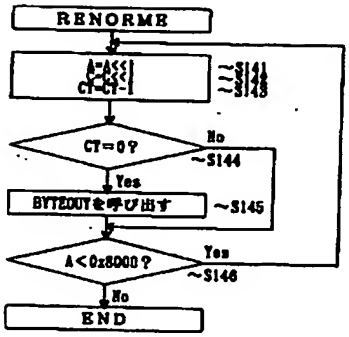
【図４７】



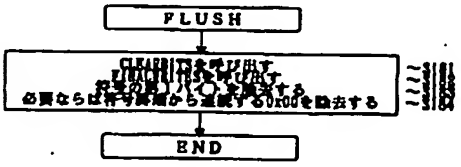
【図４８】



【図５０】



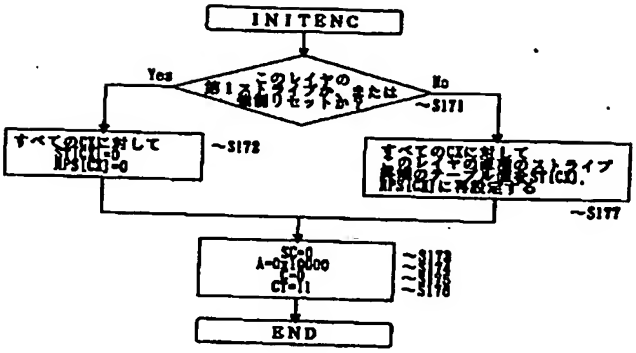
【図５３】



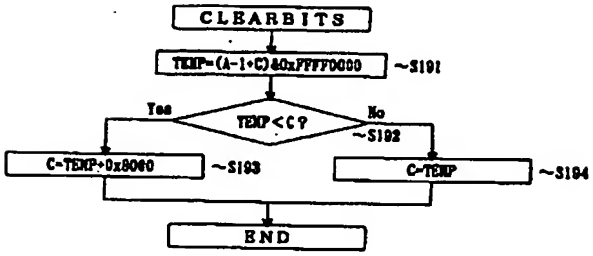
【図４４】

ST	LSZ	NLPS	RLPS	SWTCH	ST	LSZ	NLPS	RLPS	SWTCH
0	0x5a1d	1	1	1	57	0x01a4	55	58	0
1	0x2586	14	2	0	58	0x0160	56	59	0
2	0x1114	16	3	0	59	0x0125	57	60	0
3	0x080b	18	4	0	60	0x00f6	58	61	0
4	0x03d8	20	5	0	61	0x00cb	59	62	0
5	0x01da	23	6	0	62	0x00ab	61	63	0
6	0x00e5	25	7	0	63	0x008f	61	32	0
7	0x006f	28	8	0	64	0x5b12	65	65	1
8	0x0036	30	9	0	65	0x4d04	80	66	0
9	0x001a	33	10	0	66	0x412c	81	67	0
10	0x000d	35	11	0	67	0x37d8	82	68	0
11	0x0006	9	12	0	68	0x2fa8	83	69	0
12	0x0003	10	13	0	69	0x293c	84	70	0
13	0x0001	12	13	0	70	0x2379	86	71	0
14	0x5a7f	15	15	1	71	0x1edf	87	72	0
15	0x3f25	16	16	0	72	0x1a9d	87	73	0
16	0x2cf2	18	17	0	73	0x174a	72	74	0
17	0x207c	19	18	0	74	0x1424	73	75	0
18	0x17b9	40	19	0	75	0x119c	74	76	0
19	0x1182	42	20	0	76	0x0f6b	74	77	0
20	0x0cef	43	21	0	77	0x0d51	75	78	0
21	0x09a1	45	22	0	78	0x0bb6	77	79	0
22	0x0727	46	23	0	79	0x0a40	77	48	0
23	0x055c	48	24	0	80	0x5812	80	81	1
24	0x0406	49	25	0	81	0x4d1a	88	82	1
25	0x0303	51	26	0	82	0x418a	89	83	0
26	0x0240	52	27	0	83	0x3bdd	90	84	0
27	0x01b1	54	28	0	84	0x34aa	91	85	0
28	0x0144	56	29	0	85	0x2aaa	92	86	0
29	0x00f4	57	30	0	86	0x299a	93	87	0
30	0x00b7	59	31	0	87	0x2516	86	71	0
31	0x008a	60	32	0	88	0x5570	88	89	1
32	0x0068	62	33	0	89	0x4ca9	95	90	1
33	0x004e	63	34	0	90	0x44d9	96	91	0
34	0x003b	32	35	0	91	0x3a22	97	92	0
35	0x002c	33	9	0	92	0x3826	98	93	0
36	0x5a81	37	37	1	93	0x32b4	99	94	0
37	0x484c	64	38	0	94	0x2e17	93	86	0
38	0x3a0d	65	39	0	95	0x56a8	95	96	1
39	0x2e31	67	40	0	96	0x4f46	101	97	0
40	0x261f	68	41	0	97	0x47c9	102	98	0
41	0x1f33	69	42	0	98	0x41af	103	99	0
42	0x19a8	70	43	0	99	0x3a14	104	100	0
43	0x1518	727	44	0	100	0x375a	99	93	0
44	0x1177	73	45	0	101	0x5231	105	102	0
45	0x0e74	74	46	0	102	0x4e0f	106	103	0
46	0x0bfb	75	47	0	103	0x4619	107	104	0
47	0x09f8	77	48	0	104	0x415e	103	99	0
48	0x0861	78	49	0	105	0x5617	105	106	0
49	0x0706	79	50	0	106	0x50e7	108	107	0
50	0x05cd	48	51	0	107	0x4b85	109	103	0
51	0x04de	50	52	0	108	0x5597	110	109	0
52	0x040f	50	53	0	109	0x504f	111	107	0
53	0x0363	51	54	0	110	0x5a1d	110	111	1
54	0x02d4	52	55	0	111	0x5522	112	109	1
55	0x025c	53	56	0	112	0x59ab	112	111	1
56	0x01f8	54	57	0					

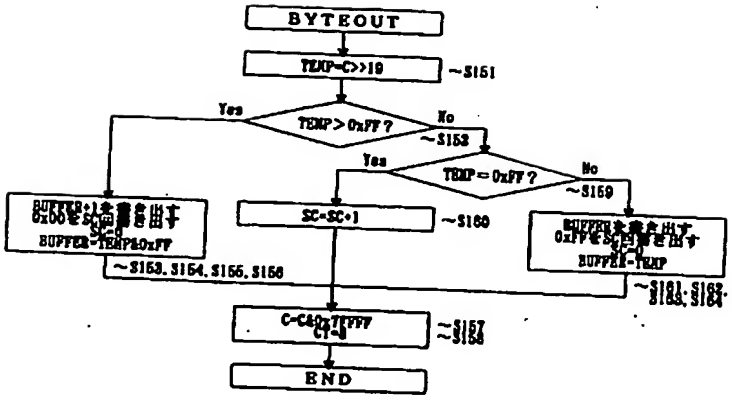
【図５２】



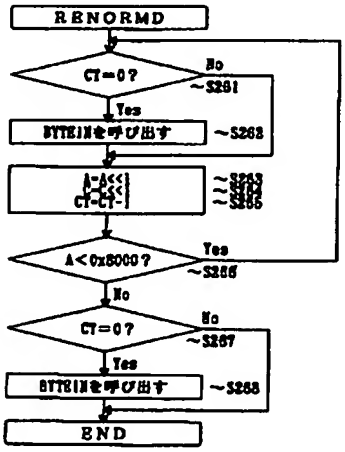
【図５４】



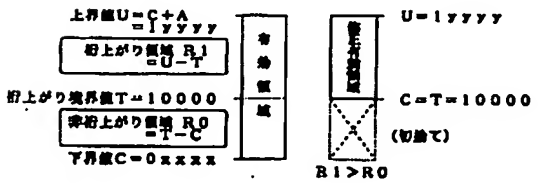
【図５１】



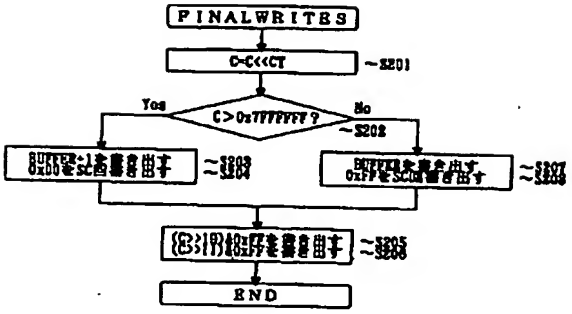
【図６０】



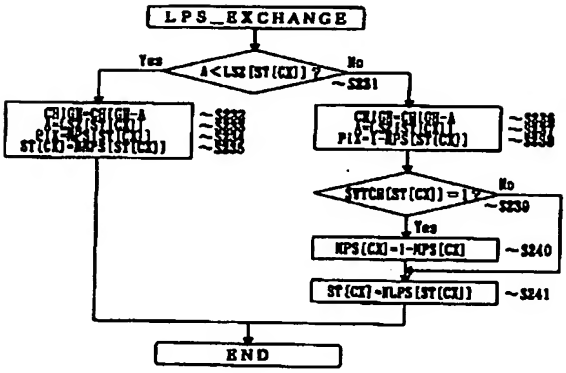
【図６３】



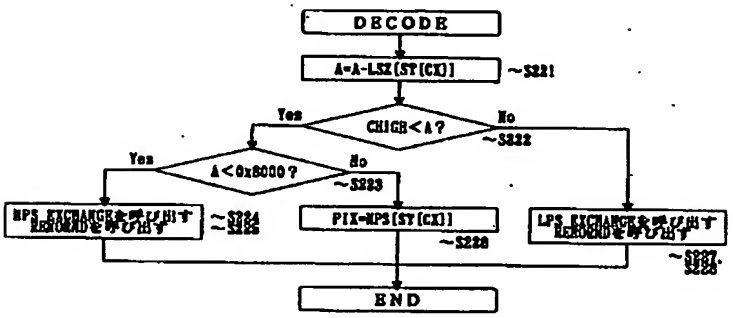
【図５５】



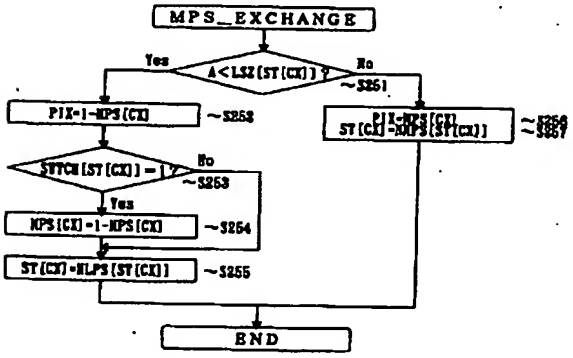
【図５８】



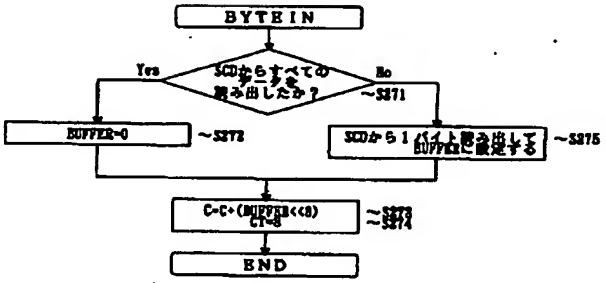
【図５７】



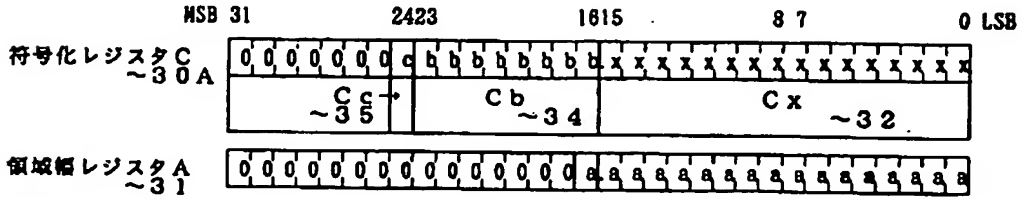
【図５９】



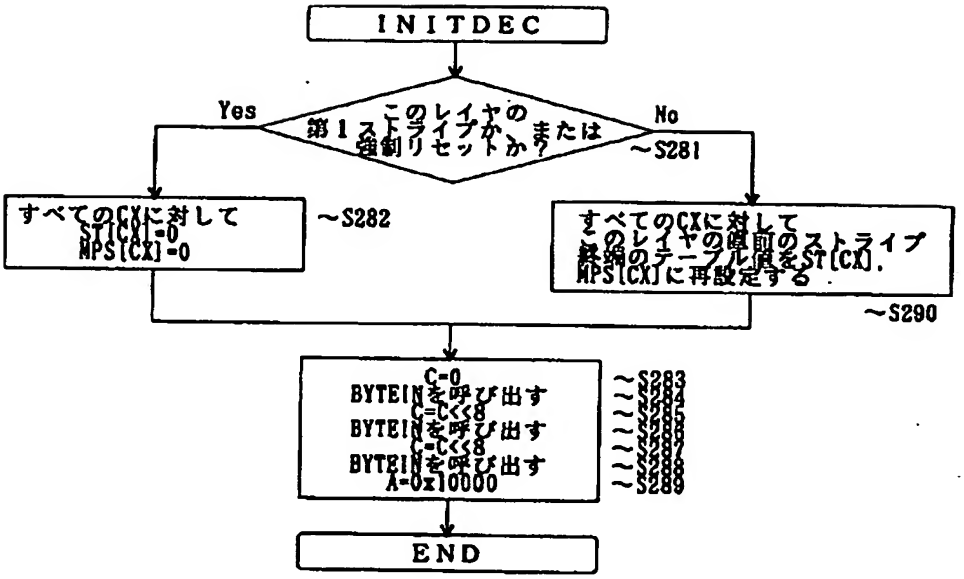
【図６１】



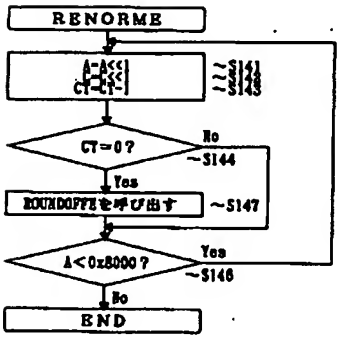
【図６６】



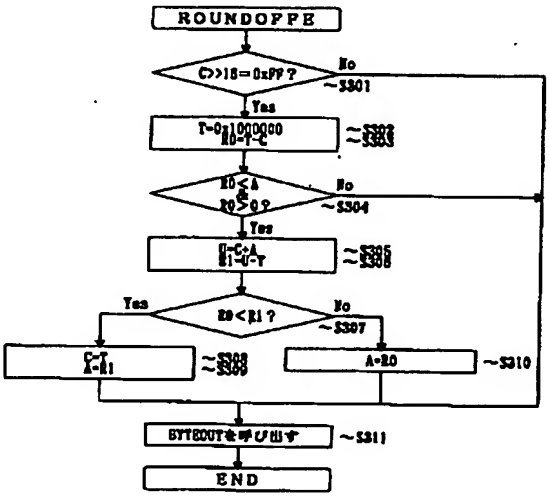
【図６２】



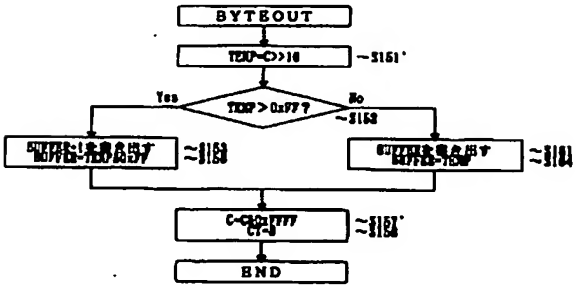
【図６７】



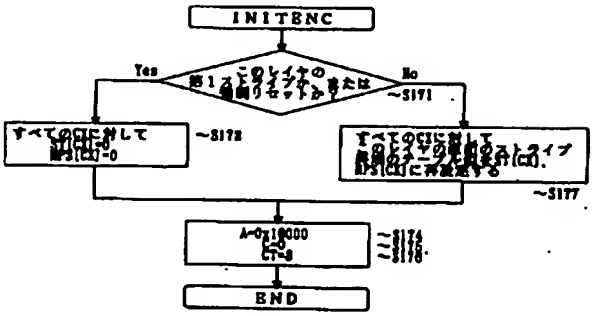
【図６８】



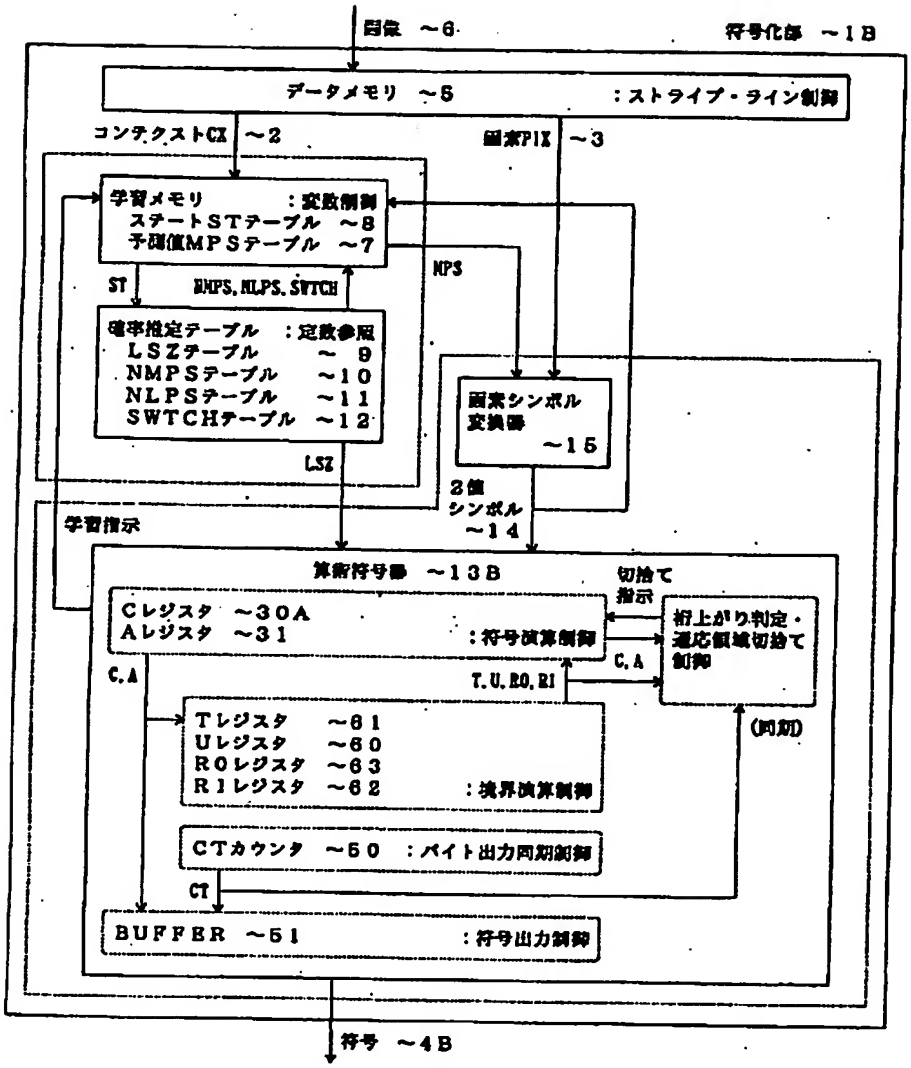
【図６９】



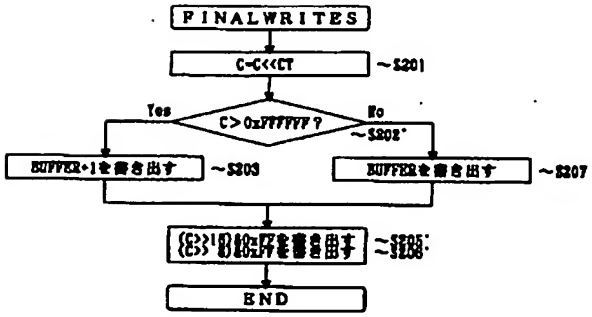
【図７０】



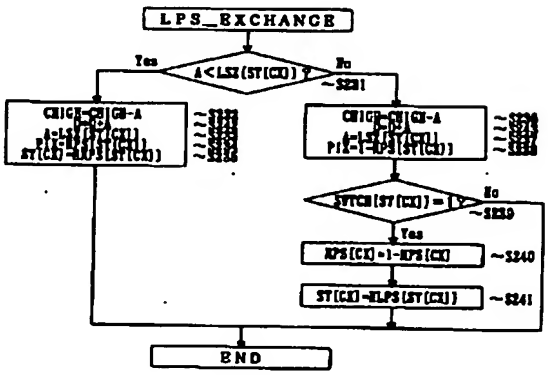
【図６４】


画像 ～6
符号化部 ～1B
データメモリ ～5 ：ストライプ・ライン制御
コンテクストCX ～2
画素PIX ～3
学習メモリ ：変数制御
ステートSTテーブル ～8
予測値MPSテーブル ～7
MPS
ST
NMPS, NLPS, SWTCH
確率推定テーブル ：定数参照
LSZテーブル ～ 9
NMPSテーブル ～10
NLPSテーブル ～11
SWTCHテーブル ～12
LSZ
画素シンボル変換器 ～15
2値シンボル ～14
学習指示
算術符号器 ～13B
切捨て指示
Cレジスタ ～30A
Aレジスタ ～31 ：符号演算制御
桁上がり判定・退応領域切捨て制御
C, A
T, U, R0, R1
(同期)
Tレジスタ ～61
Uレジスタ ～60
R0レジスタ ～63
R1レジスタ ～62 ：境界演算制御
CTカウンタ ～50 ：バイト出力同期制御
CT
BUFFER ～51 ：符号出力制御
符号 ～4B

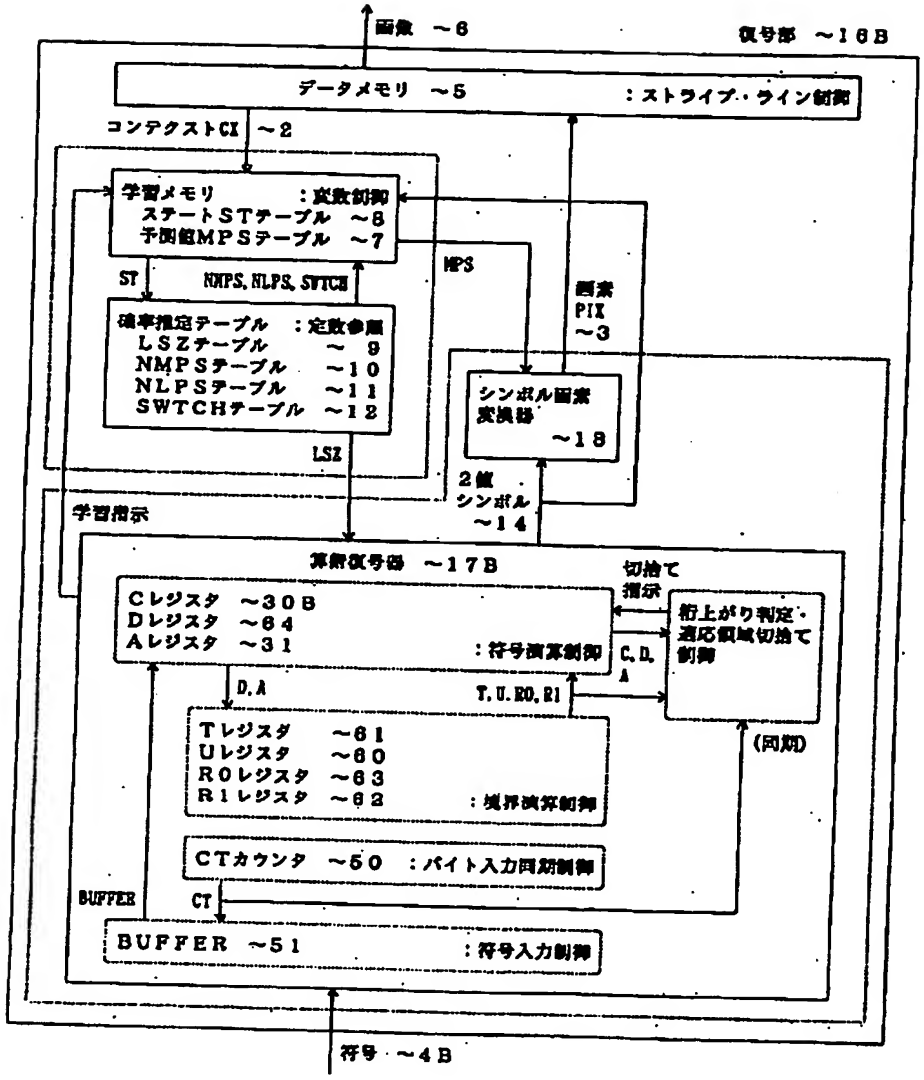
【図７１】


FINALWRITES
C=C<<CT
～S201
Yes
C>0xFFFFFF?
No
～S202
BUFFER+1を書き出す
～S203
BUFFERを書き出す
～S207
END

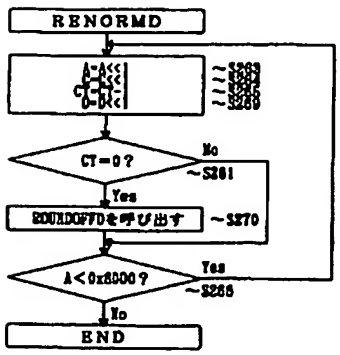
【図７２】


LPS_EXCHANGE
Yes
A<LSZ(ST(CX))?
No
～S231
END

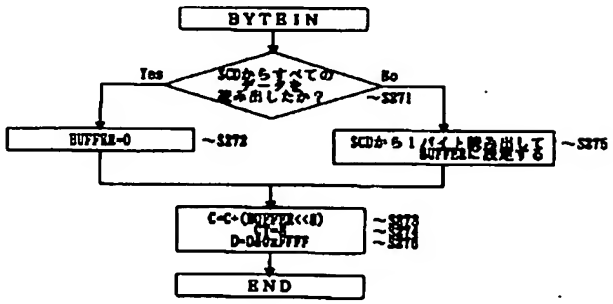
【図６５】


画像　～６
復号部　～１６Ｂ
データメモリ　～５　：ストライプ・ライン制御
コンテクストCX　～２
学習メモリ　：変数制御
ステートSTテーブル　～８
予測値MPSテーブル　～７
MPS
ST
NMPS, NLPS, SWTCH
確率推定テーブル　：定数参照
ＬＳＺテーブル　～９
ＮＭＰＳテーブル　～１０
ＮＬＰＳテーブル　～１１
ＳＷＴＣＨテーブル　～１２
LSZ
画素
PIX
～３
シンボル画素
変換器
～１８
２値
シンボル
～１４
学習指示
算術復号器　～１７Ｂ
切捨て
指示
Ｃレジスタ　～３０Ｂ
Ｄレジスタ　～３４
Ａレジスタ　～３１
：符号演算制御
C, D, A
桁上がり判定・
適応領域切捨て
制御
D, A
T, U, R0, R1
Ｔレジスタ　～６１
Ｕレジスタ　～６０
Ｒ０レジスタ　～６３
Ｒ１レジスタ　～６２
：境界演算制御
(同期)
ＣＴカウンタ　～５０　：バイト入力同期制御
BUFFER
CT
ＢＵＦＦＥＲ　～５１　：符号入力制御
符号　～４Ｂ

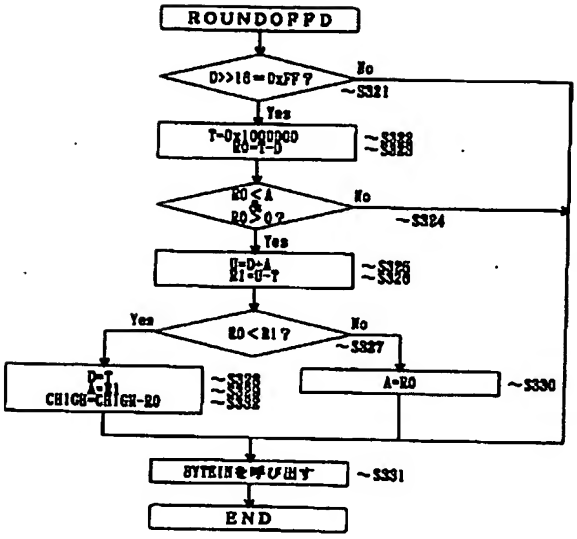
【図７３】


RENORMD
～S269
～S267
～S268
CT=0?
No
～S281
Yes
BOUNDOFFDを呼び出す
～S270
A<0x8000?
Yes
～S285
No
END

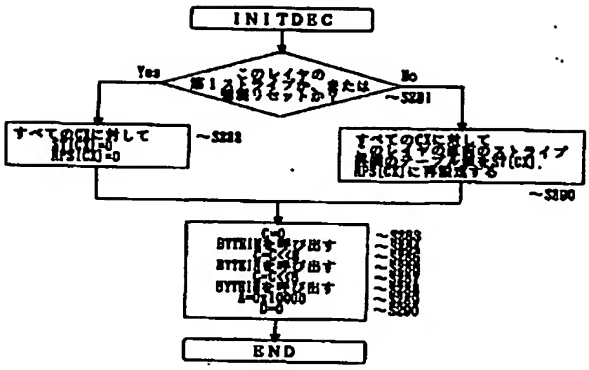
【図７５】


BYTEIN
Yes
SCDからすべての
データを
読み出したか？
No
～S271
BUFFER=0
～S272
SCDから１バイト読み出して
BUFFERに設定する
～S275
～S273
～S274
～S276
END

【図７４】



【図７６】



フロントページの続き

(72)発明者　小野　文孝
東京都千代田区丸の内二丁目２番３号　三菱電機株式会社内

Ｆターム(参考)　5C059 KK15 ME05 ME11 UA02 UA05 UA34 UA38
5C078 BA21 CA31 DA00 DA01 DA02 DA06 DA11
5J064 AA02 BA08 BA10 BB09 BB12 BC01 BC02 BC04 BC14 BC17 BC28
9A001 BB03 EE04 HH05 HH27 JJ71 KK54